Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Katedra informačních technologií

Metodika integračních projektů a jejich charakteristika

doktorská disertační práce

Doktorand : Ing. Jan Karas

Školitel : Prof. Ing. Jaroslav Jandoš, CSc.

Obor : Informatika

karas@vse.cz

Praha, červen, 2006

Prohlášení

Prohlašuji, že doktorskou práci na téma „Metodika integračních projektů a jejich charakteristika" jsem vypracoval samostatně. Použitou literaturu a podkladové materiály uvádím v přiloženém seznamu literatury.

V Praze dne 26. června 2006

……

Podpis

Abstrakt

Tato práce zpracovává problematiku interních integračních projektů, jejichž podstatnou náplní je integrace podnikových aplikací uvnitř podniku. Vysoký podíl (60-90 procent) integračních projektů je neúspěšných z pohledu rozpočtu, časového plánu nebo nesplněných požadavků. Cílem práce je nalezení specifických charakteristik vnitropodnikových integračních projektů. Jako srovnávací etalon byl použit obecný projekt implementace informačního systému. Odvozeným cílem disertační práce je z poznání integračních projektů sestavit specifickou metodiku pro implementaci integrační platformy a pro potřeby metodiky navrhnout efektivní postupy a modely.

Vědecký přínos je spatřován v detailní charakteristice integračních projektů, která fixuje obsah a terminologii těchto projektů a umožňuje komparativní analýzu s jinými IT projekty. Zároveň tato charakteristika umožňuje sestavení specifické metodiky pro integrační projekty, jejíž prvky mohou být aplikovány i do komplexních projektů vývoje IS. Metodika samotná rozšiřuje rodinu metodik, které se zaměřují na konkrétní oblasti IT, a jako taková navrhuje specifické přístupy a postupy pro integrační projekty. Metodika vznikala a byla zpětně ověřována na 4 rozsáhlých komerčních integračních projektech.

Práce je na pomezí dvou tradičních oblastí - projektového řízení a integračních technologií. V oblasti projektového řízení vychází práce z obecné metodiky MMDIS (Multidimensional Management and Development of Information System), z níž autor přebírá životní cyklus a dimenze projektu, a metodického rámce MeFIS (Methodical Framework for Information Systems Development), který metodiku zařazuje mezi další specifické metodiky projektů IS (např. ERP, CRM a BI). Z oblasti integračních technologií pokrývá průřezově celou škálu používaných integračních technologií současnosti.

Práce je rozdělena do několika navazujících celků. V první části je provedeno vymezení oblasti zkoumání, definování cílů a přínosů práce a mapování aktuálního stavu ve světě (ve formě podobně zaměřených výzkumných projektů a strukturovaným přehledem relevantních informačních zdrojů z oblasti integračních technologií a projektových metodik). Zvláštní pozornost je věnována výběru metodiky MMDIS jako mateřské metodiky. Následuje obecná charakteristika integračních projektů a jejich kategorizace navržená autorem.

Druhá část mapuje životní cyklus a dimenze integračního projektu a definuje projektový tým. Třetí část detailně zkoumá odlišnosti v integračním projektu, počínaje sestavením business plánu a specifikací integračních požadavků. Fáze analýzy je založena na abstrakci integračního procesu a integračních případů, prostřednictvím kterých se přechází do fáze návrhu.

Rozsáhlý přehled integračních postupů (tento termín pokrývá integrační koncepce, přístupy, technologie a produkty) je kategorizován podle typu integračních rozhraní a podle typu komunikace (přímé či zprostředkované). Toto členění je využito pro odhalení vzájemných souvislostí mezi integračními postupy a prakticky je využito v rozhodovacím stromu, který stanovuje, podle kterých kritérií je pro konkrétní integrační případy vybírán vhodný integrační postup.

Autor dále navrhuje model komplexní integrované architektury IS, na kterém popisuje vzájemnou spolupráci klíčových integračních komponent a hlavní a alternativní integrační cesty z pohledu architekta. Tento model je doplněn o dílčí funkční modely BPI (Business Process Integration) serveru (který je rovněž použit pro popis procesu výběru integrační platformy) a EIP (Enterprise Information Portal). Rozšířená verze modelu zahrnuje externí BRE (Business Rules Engine), kde jsou nastíněny možnosti spolupráce s ostatními sdílenými komponentami.

Práce nejde z důvodu rozsahu ve fázi návrhu až na úroveň detailního modelování v integračních technologiích, tato část je řešena odkazy na dostupné informační zdroje.

Třetí část je zakončena zkoumáním procesních a obsahových odlišností procesu testování v integračních projektech.

Závěrečná část shrnuje klíčové poznatky práce, poskytuje přehled cca 30 klíčových termínů, 100 použitých zkratek a 330 relevantních informačních zdrojů. V příloze je provedeno zmapování integračních produktů navázaných na struktury prezentované v práci a stručná charakteristika reálných integračních projektů, které byly využity při zpracování práce.

Klíčová slova: integrace podnikových aplikací, metodika, MMDIS, MeFIS, projektové fáze

Abstract (English)

This thesis investigates problems of internal integration projects where the essential content is the enterprise application integration inside the enterprise. High proportion (60-90 percents) of the integration projects is unsuccessful from budget, timetable or unfulfilled requirements point of view. The aim of the thesis is to identify the specific characterization of integration projects inside the business. The common project of the information system implementation has been chosen as a comparative etalon. The secondary aim of this thesis is to compose the specific methodology for the implementation of an integration platform and to suggest the effective methods and models for the requirements of the methodology based on the integration projects understanding.

Academic benefit is seen in the detail characterization of integration projects, that fixates the content and the terminology of these projects and that enables a comparative analysis with other IT projects. At the same time this characteristics also enables to compose the specific methodology for integration projects, whose parts can be adopted into the complex projects of IS development. Methodology itself enlarges the family of methodologies that are focused on certain areas of IT and it designs the specific approaches and methods for integration projects. The methodology arose from, and has been retrospectively verified on, four large commercial integration projects.

The thesis is on a border of two traditional areas - project management and integration technologies. In the area of project management the thesis comes from a common methodology MMDIS (Multidimensional Management and Development of Information System), from which the author takes over the lifecycle and the dimensions of a project, and MeFIS (Methodical Framework for Information Systems Development), that classifies the methodology among other specific project methodologies of IS. (e.g. ERP, CRM and BI). From the area of integration technologies the thesis cross-sectionally covers whole spectrum of the present day used integrative technologies.

The thesis is divided into the several connected units. In the first part it is defined the scope of the research, the goals, the benefits of the thesis and there is mapped the current state of knowledge in the world as well (presented by related research projects and brief survey of informational sources in the area of integration technologies and project methodologies). Particular attention is dedicated to the selection of the methodology MMDIS as motherly methodology. It continues with the common characterization of integration projects and their categorization proposed by author.

The second part maps the lifecycle and dimensions of integration project and it defines the project team. The third part investigates the differences within the integration project, starting by the composition of the business case and by the specification of integration requirements. The phase of analysis is based on the abstraction of integration process and on the integration cases, by means of them it is moved on the phase of design.

Extensive overview of integration methods (this term covers integration concepts, approaches, technologies and products) is categorized according to the type of integration interfaces and according to the type of communication (direct or mediate). This categorization is used for the detection of mutual dependencies among the integration methods and it is practically used in the decision tree, that defines according which criteria is chosen the suitable integration methods.

The author suggests a model of complex integration architecture of IS, on that he describes the mutual cooperation of the key integration components and main and alternative integration ways from the architect point of view. This model is completed by the partial functional models of BPI (Business Process Integration server), which is also used for the description of the process of the integration platform selection, and EIP (Enterprise Information Portal). The enlarged version of the model includes the external BRE (Business

Rules Engine), where are demonstrated the possibilities of the cooperation with the others shared components.

The thesis does not cover the detail level of modeling in integration technologies due to wide range. This part is solved by external links to the accessible informational sources.

The third part is finished with the investigation of procedural and content differences of the process of testing within the integration projects.

The final part summarizes the key findings of the thesis, it provides the overview of approximately 30 key terms, 100 used abbreviations and 330 relevant sources of information. The annexes provide the overview of integration products mapped to the categories presented in the thesis and the brief characterization of real integration projects, which were used to process and to verify the thesis.

Keywords: enterprise application integration, methodology, MMDIS, MeFIS, project phases

Abstrakt (Deutsch)

Diese Arbeit befasst sich mit der Problematik von internen Integrationsprozessen, deren wesentlicher Inhalt eine innenbetriebliche Integration von betrieblichen Applikationen ist. Ein hoher Anteil (60-90%) der Integrationsprojekte scheitert an Budget, Zeitplan oder Nichterfüllung der Anforderungen. Das Ziel der Arbeit ist die Findung von spezifischen Charakteristika von innenbetrieblichen Integrationsprojekten. Als vergleichendes Bezugsnormal dient ein allgemeines Projekt zur Implementierung eines Informationssystems. Das abgeleitete Ziel der Arbeit ist es, eine spezifische Methodik für Implementierung der Integrationsplattform zu erstellen und für die Bedürfnisse der Methodik effiziente Methoden und Modelle vorzuschlagen.

Ein wissenschaftlicher Beitrag wird in einer detaillierten Charakteristik von Integrationsprojekten gesehen, die den Inhalt und die Terminologie von diesen Projekten fixiert und ermöglicht eine komparative Analyse mit anderen IT- Projekten und ermöglicht gleichzeitig die Schaffung einer spezifischen Methodik für die Integrationsprojekte, deren Elemente auch in komplexen Projekten IS-Entwicklung angewendet werden können. Allein die Methodik erweitert die Methodikfamilie, die sich auf konkrete IT- Gebiete konzentriert und schlägt konkrete spezifische Herangehensweise und Methoden für die Integrationsprojekte vor. Die Methodik entstand und wurde auf Grund von 4 umfangreichen kommerziellen Integrationsprojekten überprüft. Die Arbeit bewegt sich an der Grenze von zwei traditionellen Gebieten – Projektsteuerung und Integrationstechnologien. Auf dem Gebiet der Projektsteuerung kommt die Arbeit von der allgemeinen MMDIS- Methodik (Multidimensional Management and Development of Information System) aus, von der der Autor den Lebenszyklus und die Projektdimension übernimmt und von dem methodischen Rahmen des MeFIS (Methodical Framework for Information Systems Development) aus, der die Methodik zu anderen spezifischen Methoden (z.B. ERP, CRM und BI) zuordnet. Aus dem Gebiet der Integrationstechnologien stellt die Arbeit eine ganze Skala von zur Zeit genutzten Technologien dar.

Die Arbeit ist in einige an sich anschließende Teile gegliedert. Im ersten Teil wurde die Abgrenzung des Forschungsgebietes definiert, des weiteren die Ziele und der Beitrag der Arbeit sowie die Feststellung des aktuellen Standes weltweit (in Form von ähnlich gezielten Forschungsprojekten und einem strukturierten Überblick über die relevanten Informationsquellen aus dem Gebiet der Integrationstechnologien und Projektmethodik).

Eine besondere Aufmerksamkeit wurde der Auswahl der MMDIS als Muttermethodik gewidmet. Es folgt eine allgemeine Charakteristik von Integrationsprojekten und deren Kategorisierung, die vom Autor vorgeschlagen wurde.

Im zweiten Teil wird der Lebenszyklus und die Dimension eines Integrationsprojektes dargestellt und ein Projektteam definiert. Im dritten Teil werden die Unterschiede im Integrationsprojekt detailliert untersucht, begonnen mit Erstellung eines Businessplanes und Spezifizierung der Integrationsanforderungen. Die analytische Phase basiert auf einem Abstraktionsprozess und der Integrationsfälle, mittels deren man zu der Vorschlagsphase kommt. Ein umfangreicher Überblick über Integrationsmethoden (dieser Termin ist ein Oberbegriff für Integrationskonzepte, Herangehensweise, Technologien und Produkte) ist kategorisiert nach dem Typ der Integrationsgrenze und der Kommunikation (direkte oder vermittelte). Diese Aufteilung ist für die Entdeckung der gegenseitigen Zusammenhänge zwischen Integrationsmethoden und dies ist praktisch im Entscheidungsbaum genutzt, in dem festgelegt wird, nach welchen Kriterien eine geeignete Methode für konkrete Integrationsfälle gewählt wird.

Des Weiteren schlägt der Autor ein Modell einer komplexen integrierten IS-Architektur, wo er die Zusammenarbeit von Schlüsselkomponenten der Integration und die primären und alternativen Integrationswege aus der Sicht des Architekten beschreibt. Dieses Modell wird um partielle Funktionsmodelle des BPI (Business Process Integration Servers)

(der auch für die Beschreibung der Auswahl einer Integrationsplattform benutzt wird) und des EIP (Enterprise Information Portal) ergänzt. Die erweiterte Version umfasst eine externe BRE (Business Rules Engine), wo die Möglichkeiten für eine Zusammenarbeit mit anderen Gemeinschaftskomponenten dargestellt sind.

Die Arbeit geht in der Vorschlagsphase nicht - aus Umfangsgründen- in eine detaillierte Simulation in den Integrationstechnologien, dies wird durch Hinweise auf die zugänglichen Informationsquellen gelöst.

Der dritte Teil wird mit der Untersuchung von Prozess- und Inhaltsunterschieden des Testprozesses in den Integrationsprojekten abgeschlossen.

Der abschließende Teil fasst die Schlüsselerkenntnisse der Arbeit zusammen, bietet einen Überblick über ca. 30 Schlüsseltermine, 100 verwendete Abkürzungen, 330 relevante Informationsquellen. Im Anhang wurde eine Darstellung von Integrationsprodukten, die an die in der Arbeit präsentierten Strukturen angebunden sind und eine Kurzcharakteristik der realen Integrationsprojekten, die bei der Erarbeitung dieser Arbeit verwendet wurden.

Die Schlüsselwörter: Integration von betrieblichen Applikationen, Methodik, MMDIS, MeFIS, Projektphasen

Obsah

Přehled obrázků

Přehled tabulek

1. Úvod

Úvodní kapitola vymezuje problematiku vnitropodnikové integrace aplikací jako oblast zkoumání této práce. Zmiňuje dvě příbuzné oblasti B2B a EIP. V další podkapitole formuluje problém velkého procenta neúspěšnosti integračních projektů a pomocí dvou pracovních hypotéz navrhuje řešení zvýšení efektivnosti integračních projektů cestou nalezení specifických rysů integračních projektů a vytvořením metodiky pro tento typ projektů. V dalších podkapitolách definuje vědecké a praktické přínosy této práce, postup zpracování práce včetně použitých vědeckých metod a strukturu práce. V poslední podkapitole shrnuje současný stav problematiky ve světě, nejdříve v oblasti projektového řízení, následně v oblasti integračních technologií a na závěr uvádí výzkumné projekty zaměřující se na metodiky integračních projektů.

1.1 Oblast zkoumání

Pro svou disertační práci jsem si vybral oblast informačních technologií, která se zabývá **vnitropodnikovou integrací aplikací**. Tato oblast se označuje EAI (Enterprise Application Integration). Problematiku integrace jsem zvolil především proto, že se jí již několik let teoreticky i prakticky zabývám na pozici konzultanta a architekta IT řešení. Tato pozice mi umožňuje identifikovat otevřené otázky při realizaci EAI a zároveň hledat a navrhovat jejich možná řešení, která jsou následně ověřována v praxi.

Vymezit zkoumanou oblast není v tomto případě zcela jednoduché, neboť se už ze své podstaty prolíná s dalšími oblastmi IT. Dle [LAND, 2004] definuje IEEE Standard Glossary of Software Engineering Terminology obecnou integraci jako "proces kombinování softwarových komponent, hardwarových komponent, nebo obou, do společného systému".

Integraci rozdělujeme (dle [JANDOŠ, 2004]) dle automatizace na ruční (např. přenosem dat na médiu) a automatizovanou. Tato práce se zabývá integrací automatizovanou. V informačních zdrojích existuje řada definic EAI, z nichž vybírám dvě následující.

[BASS, 2002] definuje EAI jako "systematický přístup k problému integrace aplikací, který nabízí řešení, které umožňuje řídit množství komplexních transakcí skrze všechny podnikové procesy napříč různým platformám a systémům." Za nejdůležitější slovo v této definici považuji slovo "přístup", což nám říká, že EAI není jen o konkrétním nástroji či technologii, ale o celé koncepci řešení. Úlohou EAI je za pomoci informačních technologií podporovat návaznost interních podnikových procesů.

[LOSAVIO, 2003] definuje EAI široce jako "plány, metody a nástroje zaměřené na modernizování, konsolidaci, integraci a koordinaci IS (informačního systému) uvnitř podniku, kde hrají standardy důležitou roli".

K EAI existují dvě obsahově velmi příbuzné oblasti. První z nich je **B2B** (Business-to-Business) integrace, jejíž úkol je shodný s EAI, avšak týká se externích podnikových procesů (tedy procesů, které navazují na externí prostředí podniku) ve vztahu k partnerům podniku. Vzhledem k podobnosti obou úloh EAI a B2B integrace se často používají stejné nebo odvozené technologie, postupy i integrační koncepce, ale liší se důrazem na jiné aspekty integrace. B2B a A2A (Application-to-Application, integrace aplikací ve smyslu EAI) spolu dokáží úzce spolupracovat (více viz [BUSSLER, 2003] a [BUSSLER, 2002a]). Implementovaná EAI je nutným předpokladem pro B2B ([ERASALA, 2003]). V průzkumu [CLAY] odpovědělo 37,5% respondentů, že jejich metodika pro integraci aplikací je plně integrovaná s metodikou pro B2B. [BUSSLER, 2002b] spatřuje odlišnosti mezi B2B a A2A v tom, že A2A:

- nemá standardizované komunikační protokoly,
- vztah mezi aplikacemi není řízen právní smlouvou,

- komunikace je řízena centrálně integračním serverem (pozn. autora: toto platí pro uspořádání typu hub, uspořádání typu bus pracuje na bázi decentralizovaného řízení),
- předávaná data mohou vyžívat interní reference mezi aplikacemi,
- převodní formátování dat mezi aplikacemi se provádí na integračním serveru,
- není třeba právně vymezovat, kdy jsou data uvnitř a kdy vně podniku,
- výskyt nekonzistentního stavu se řeší interně, nikoli soudem nebo arbitráží,
- pro adresování (kontaktování) aplikací se využívá širší množiny technologií než v B2B,
- zabezpečení předávaných dat může být nižší úrovni, neboť probíhá uvnitř podniku.

Druhou blízkou oblastí EAI je **EIP** (Enterprise Information Portal), která se zabývá agregací informací a služeb podniku pro zákazníky (B2C - Business-to-Customer), partnery a pro zaměstnance (B2E - Business-to-Employee). Styčnou plochu mezi EAI a EIP nalézáme v potřebě EIP přistupovat k interním systémům, což řeší právě EAI.

[JURIC, 2002] označuje takto vymezenou oblast EAI jako intraEAI, zatímco B2B integraci jako interEAI. V této práci se budeme zabývat interní EAI, o dalších zmíněných oblastech se zmíním pouze okrajově, neboť by přesáhly rozsah této práce a nejsou relevantní k jejím vytčeným cílům.

EAI bývá v užším pojetí chápáno jako integrace v reálném čase na aplikační úrovni pro přímou podporu podnikových procesů. Tomuto pojetí se nijak nebráním, avšak rozhodl jsem se rozšířit záběr této práce i na dávkovou integraci (kde se zpravidla jedná o časově zpožděnou integraci). Důvodem pro toto rozhodnutí je fakt, že se jedná o přístupy, které se navzájem ovlivňují a zároveň si konkurují. Za rozhodující kritérium, zda některou koncepci či technologii do této práce zařadit, jsem si určil kladnou odpověď na otázku, zda umožňují integraci interních podnikových procesů.

Z hlediska vztahu aplikací rozeznáváme integraci (dle [JANDOŠ, 2004]):

- "těsnou, kdy jednotlivé aplikace jsou součástí integrovaného programového komplexu (balíku – např. ERP),
- volnou, kdy jednotlivé aplikace jsou navzájem nezávislé (často zpracovávané na různých geograficky rozmístěných informačních systémů – tj. tyto informační systémy tvoří distribuovaný systém)."

Tato práce se věnuje pouze **integraci s volnou vazbou heterogenních aplikací**.

Mimo rozsah této práce jsem rovněž stanovil technologie, které jsou navržené pouze proprietárně pro propojení jen omezené skupiny systémů. Stejně tak technologie, které realizují integrační vrstvu uvnitř jednoho systému/ produktu (tzv. integrované balíky typu SAP). Důvodem pro toto omezení je praktická nemožnost byť jenom zmapovat všechny tyto technologie a rovněž jejich zhodnocení může mít jen omezený přínos a využitelnost.

Z hlediska řízení projektu není zvláštní důraz kladen na rozdíly mezi realizací projektu uvnitř firmy (in-house přístup) a vně firmy (dodavatelský přístup), a to ani v době před ani po zavedení systému. Prezentované závěry jsou v zásadě aplikovatelné pro oba tyto přístupy a neřeší specifika outsourcingových projektů. Specifická problematika integrace u **ASP** (Application Service Provider) je rovněž mimo rozsah této práce, neboť je na rozhraní mezi B2B a A2A (více viz [BUSSLER, 2003]). Problematikou integrace ASP do PIS se zabývá rovněž [VOŘÍŠEK, 2004], který pro malé a střední podniky doporučuje přenechat integraci do PIS na ASP (především kvůli pořizovací ceně integrační platformy).

1.2 Řešený problém

Z řady empirických studií vyplývá, že integrační projekty jsou ve většině případů neúspěšné. [CRAGGS, 2005] uvádí s odkazem na průzkum Integration Consortia, že 70 procent integračních projektů selže vzhledem k rozpočtu, termínům či kvalitě. [KING, 2004] uvádí s odkazem na průzkum společnosti Standish Group, že selhává 95 procent integračních projektů. [BARLAS, 2003] cituje Steva Craggse z EAI Industry consortia

(předchůdce Integration Consortia), který říká, že selhává 75 procent integračních projektů, příčinou selhání není technologická složitost, ale manažerské chyby v plánování a realizaci projektů. [CHAPPELL, 2004] a [FARGES, 2003] s odvoláním na report "Reducing Integration Costs" od společnosti Forrest z konce roku 2001 zmiňuje, že méně než 35 procent projektů končí v požadovaném čase a nákladech a průměrná délka integračního projektu je více než 20 měsíců.

Integrační problémy ovšem nejsou záležitostí pouze čistě integračních projektů, ale např. i projektů implementace ERP, jejichž hlavním problémem (82 procent projektů) je podle [THEMISTOCLEOUS, 2001a] (a [THEMISTOCLEOUS, 2001b]) právě integrace se stávajícími systémy. Obecně můžeme za neúspěšný projekt považovat takový, kde reálný rozdíl přínosů a nákladů neodpovídá očekávanému rozdílu a je oproti očekávání menší. K tomu dochází v důsledku dodatečných nákladů (např. v důsledku zdržení při realizaci projektu) nebo nedostatečným přínosům (např. v důsledku jiné funkcionality implementovaného produktu). O existenci tohoto problému svědčí řada výzkumných projektů, které jsou zmíněny v kapitole 1.6 Aktuální stav ve světě.

Potřeba metodiky pro integrační projekty je jasně formulována v [JOHANNESSON, 2000], který vytyčuje čtyři výzkumné směry v oblasti aplikační a procesní integrace:

1. Knihovny procesů - vytvoření katalogů podnikových procesů, které by mohl návrhář snadno používat při modelování procesů. Především je třeba vyřešit strukturu a vyhledávání v katalogu, kde je možné využít mechanismů dekompozice a specializace, či koordinace závislostí procesů nebo jejich komunikace.
2. **Metodika pro efektivní EAI** - metodika by měla pokrývat zejména analýzu podnikových dat, analýzu podnikových procesů a jejich návrh ve vazbě na technologie. Důležitý je procesní přístup k integraci.
3. Adaptivní řízení procesů - současné systémy pro podporu procesů, typu procesních integrátorů či worklow systémů, jsou efektivní pro předvídatelné a opakující se procesy. Nejsou však efektivní v přizpůsobování se vyjímečným stavům (typu selhání základního systému či prostředí, aplikačního selhání vedle očekávaných a neočekávaných vyjímečných stavů).
4. Přesun obchodní logiky ze systémů do procesů - směřuje k přesunu business pravidel do procesní integrační vrstvy.

K druhému výzkumnému směru, do kterého tato práce směřuje, upřesňuji, že:

- Analýzou podnikových dat se metodika zabývá při analýze a popisu integračních případů. Nesnaží se vytvořit všeobjímající datový model celého podniku, ale soustředí se primárně na zadanou integrační úlohu.
- Analýza a návrh podnikových procesů jsou delegovány na specializované projekty typu BPR, popsaná metodika se soustředí na integrační část procesů, kterou dokáže řešit integračními technologiemi. Pro přechod od procesů k technologiím je v metodice popsán konkrétní postup.

Zároveň se tato práce přímo dotýká i posledně jmenovaného výzkumného směru, když popisuje rozšířený integrační model s vazbou na BRE, což je prvek, který přímo podporuje extrakci business pravidel z aplikací.

Příčiny problému neúspěšnosti integračních projektů můžeme rozdělit do dvou skupin:

1. Obecné příčiny vyplývající z podstaty projektů, jako např. chybně definovaný projekt, nízká kvalita projektového týmu, nedostatečná podpora projektu.
2. Specifické příčiny vyplývající z nerespektování odlišností integračních projektů od obecných projektů vývoje informačních systémů.

Zde vytvářím první pracovní hypotézu, tedy, že **integrační projekty mají specifické charakteristiky** oproti obecným projektům vývoje informačních systémů. Tuto hypotézu budu v práci prokazovat postupným zkoumáním fází integračního projektu a bude prokázána, pokud se podaří nalézt alespoň jednu odlišnost, která se vyskytuje pouze u integračních projektů.

Obecným příčinám selhání projektů se v této práci věnovat nebudu, neboť se jim věnuje samostatná oblast řízení projektů.

Vycházeje z poznání integračních projektů, tak jak budu nalézat jejich specifické rysy, budu vytvářet metodiku specifickou pro integrační projekty. Tato metodika bude obsahovat nalezená specifika a bude obsahovat způsoby jejich řešení (které mohou být s rozvojem poznání v této oblasti dále optimalizovány).

Druhá pracovní hypotéza tedy říká, že **pro nalezené odlišnosti integračních projektů lze nalézt způsoby, jak je zohlednit v projektové metodice**, a tím ji učinit efektivnější. Tuto hypotézu budu v práci prokazovat vytvářením metodiky specifické pro integrační projekty a bude prokázána, pokud se pro každou nalezenou odlišnost podaří nalézt obecný postup, jak ji v projektu řešit. Tím, že bude nalezen alespoň jeden obecně aplikovatelný postup pro danou odlišnost, můžeme metodiku považovat za efektivnější, neboť zkracuje dobu projektu (a tím i náklady) o čas nutný pro hledání a verifikaci řešení v konkrétním integračním projektu (což je obecně smysl a princip projektových metodik).

Za přínos k řešenému problému můžeme považovat přehled charakteristik integračních projektů, které mohou být dále zkoumány, a specifickou metodiku pro integrační projekty, která umožní jejich efektivnější realizaci. Ve vztahu k řešenému problému **není podstatné, že jako výchozí metodika byla zvolena MMDIS, neboť charakteristika integračních projektů je na metodice nezávislá a nalezené postupy jsou popsány tak, aby je bylo možno transformovat i do jiných metodik. Stejně tak není podstatný kompletní výčet integračních postupů ani hloubka jejich popisu, neboť ten slouží pouze jako východisko pro pochopení šíře a rozmanitosti integračních problémů a jejich řešení**, z kterého následnou syntézou vzniká obecný postup pro navrhování integračních řešení.

V případné existenci jiné projektové metodiky pro integrační projekty nespatřuji překážku pro tuto práci, neboť smyslem této práce není metodika samotná, ale popsání a vysvětlení specifických rysů integračních projektů. Tato charakteristika je společným základem pro sestavení efektivních metodik pro integrační projekty a zde prezentovaná metodika je především důkazem významnosti a smysluplnosti poznávání v této oblasti.

1.3 Cíle a přínosy

Cílem této práce je nalezení specifických charakteristik vnitropodnikových integračních projektů. Srovnávacím etalonem je obecný projekt implementace informačního systému. Odvozeným cílem je z poznání integračních projektů sestavit specifickou metodiku pro implementaci integrační platformy a pro potřeby metodiky navrhnout efektivní postupy a modely.

Vědecký přínos této práce spatřuji ve třech směrech:

- Charakteristika integračního projektu poskytne solidní základ pro další zkoumání tohoto typů projektů - konkrétně lze hledat další způsoby optimalizace formou zdokonalování navržené metodiky, dále pracovat s navrženými modely a rozpracovávat je o další poznatky a návrhy; vedle navržených modelů sjednocuje charakteristika projektu odbornou terminologii, kterou lze užívat v odborných textech.
- Charakteristika, resp. metodika integračních projektů je připravena pro komparaci s dalšími specifickými metodikami pro vývoj informačních systémů (např. v oblasti Business Inteligence, Enterprise Information Portal, Customer Relationship Management,

atd.). Komparací specifickým metodik dochází nejen k ověřování metodiky vzaté jako společný etalon, ale i k jejímu prohlubování a rozšiřování díky zobecněným poznatkům ze specifických metodik. Ve vztahu k obecné metodice pro vývoj informačních systémů se tak rozrůstá rodina odvozených metodik, které postupně mohou začít tvořit kompaktní celek.

- Systémový a strukturovaný přístup k analýze obsahu integračního projektu odhaluje oblasti a souvislosti, které dosud nebyly s problematikou provázány a popsány (např. EIP). Jejich zasazení do rámce metodiky, resp. jejích modelů umožní jejich další zkoumání v nových souvislostech.

Praktický přínos očekávám následující:

- Zpracovávaná metodika bude průběžně ověřována v reálném prostředí, a bude proto aplikovatelná na reálné integrační projekty, což povede k jejich efektivnějšímu průběhu. Její praktické využití vede k efektivnějšímu řízení a vyšší úspěšnosti EAI projektů.
- Pro oblast integrace aplikací není zpracována dostupná metodika, která by zohledňovala potřeby současných integračních projektů; současné integrační projekty jsou realizovány s využitím obecných metodik pro vývoj informačních systémů, které poskytují dobrý projektový základ, ale nejsou dostatečně efektivní ve vztahu ke konkrétním projektům. Ani metodika prezentovaná v této práci není absolutně efektivní, neboť nejde (ani nemůže) na úroveň konkrétních projektů (což je způsobeno tím, že metodika je společná pro více projektů, ale každý projekt je unikátní), ale zohledňuje meziúroveň mezi obecným projektem a konkrétním projektem - určitou společnou charakteristiku skupiny projektů , která je podmnožinou obecných projektů.
- Metodika je stavěna modulárně v tom smyslu, že její ucelené části ve formě specifických postupů lze využít i relativně samostatně (např. proces výběru integrační platformy, rozhodovací strom pro výběr vhodné integrační technologie). Tato vlastnost metodiky umožňuje její části aplikovat i v projektech, v kterých je integrační úloha pouze částí celého projektu.

Ke stanovení těchto cílů mne vedla reálná absence informací o odlišnostech EAI projektů od modelových projektů vývoje informačních systémů a nedostatek utříděných informací o nabízených a používaných koncepcích a technologiích v dostupných informačních zdrojích.

V praxi se prezentované poznatky uplatní na pozicích konzultanta, analytika, designera (návrháře), architekta, projektového manažera v oblasti EAI projektů, sekundárně i na obdobných pozicích v oblastech B2B integrace či EIP. Z hlediska postavení zákazník - dodavatel se soustředí hlavně na roli dodavatele, většina informací je však zcela relevantní i pro roli zákazníka. Pro pochopení hlavních myšlenek prezentovaných v této práci je třeba, aby byl čtenář obeznámen s řízením IT projektů a měl základní znalosti integrační problematiky, v ideálním případě se alespoň částečně podílel v některé ze zmíněných rolí na IT projektu vyžadujícím integraci s dalšími interními systémy podniku.

1.4 Postup zpracování práce

S přípravou práce jsem začal v roce 2000, kdy jsem měl možnost se aktivně podílet na svém prvním integračním projektu (viz Projekt 1 - Podnikový informační portál (2000 - 2003)), kde jsem metodou pozorování získával první informace o tomto typu projektů. V této fázi jsem získával poznatky z oblasti **projektového řízení** a specializoval jsem se na technologickou oblast **integrace prezentační vrstvy** a **bezpečnosti integrovaného systému**. V souladu s tím jsem analyzoval informační zdroje s cílem zjistit rozsah funkcionality integračních nástrojů. Syntézou takto získaných informací vznikl článek [KARAS, 2002], studie na téma Porovnání produktů pro realizaci podnikových informačních portálů (76 str.) (určena výhradně pro komerční účely) a diplomová práce [KARAS, 2001] zaměřená na bezpečnost PIS. V disertační práci jsou výsledky prezentovány zejména v kapitolách 5.11 Dimenze SEC (doplněná), 12.1.10 Zabezpečení a 12.3 EIP. Funkcionalita

EIP je popsána na modelu, který jsem vytvořil zobecněním poznatků získaných syntézou informací o EIP, výsledný model jsem verifikoval oproti informačním zdrojům.

V další fázi přípravy práce jsem se věnoval zkoumání **funkcionality integračních serverů** a **využití ODS při integraci**. K praktickému pozorování jsem dostal příležitost v dalším integračním projektu (viz Projekt 2 - Integrace CRM (2004)), kde jsem získal poznatky zejména o možnostech spolupráce těchto klíčových integračních komponent (která v pozdějších fázích práce vedla k vytvoření modelu prezentovaného v kapitole 11.2 Komplexní integrační model PIS). Zároveň jsem analýzou informačních zdrojů a konzultacemi s řadou expertů na integraci, kteří se na projektu podíleli, získal znalosti o funkcionalitě integračních serverů a ODS, které jsou v disertační práci prezentovány na modelu v kapitole 12.1 Integrační server (ODS je dostatečně popsána v informačních zdrojích). Na základě tohoto modelu (resp. jeho dřívější verze) jsem publikoval článek [KARAS, 2004]. Problematice výběru integrační platformy je věnována kapitola 14 Výběr integrační platformy.

Během této fáze jsem rovněž rozvíjel formou pozorování zmíněného projektu své poznatky o řízení integračních projektů. Během **analytické fáze** projektu se začaly užívat termíny **integrační proces a integrační případ**, které jsem formalizoval v kapitole 9 Integrační proces a integrační případ. Sestavil jsem analytický popis integračního případu, který je popsán v kapitole 9.3 Analýza integračního případu, a který byl v tomto projektu experimentálně ověřen (po dohodě s kolegy v projektovém týmu). Metodický přístup k fázi analýzy a designu byl prezentován v [KARAS, 2006b].

V dalším projektu (viz Projekt 3 - Aplikace pro zřizování služby (2004, 2006)) jsem ověřil (opět formou řízeného experimentu) aplikovatelnost tohoto přístupu i **na složitých integračních úlohách** a to nejen ve fázi analýzy, ale i designu, vývoje a testování.

V závěrečné fázi přípravy této práce jsem se věnoval analýze **zbývajících integračních postupů** z informačních zdrojů a na základě syntézy získaných poznatků jsem navrhnul hlediska pro strukturování integračních postupů, které jsem následně uplatnil při sestavení mapy integračních postupů, která je prezentována v kapitole 10 Integrační postupy.

Přehled integračních postupů (viz 10.6 Přehled integračních postupů) je však pouze východiskem pro kapitolu 10.7 Rozhodovací strom integračních postupů, kde jsem popsal způsob rozhodování při volbě integračního postupu a pro kapitolu 11 Model komplexní integrované architektury PIS (resp. kapitolu 13 Rozšířený model komplexní integrované architektury PIS). Tento model představuje základnu pro další rozhodovací proces (řešící optimální integrační cesty v PIS), který je však nadřazen jednotlivým integračním projektům. Obě kapitoly vznikly na základě pozorováním činnosti integračních expertů při řešení integračních úloh, komparací jejich chování s informacemi z informačních zdrojů a rozhovorů s integračními experty a následnou explanací jejich chování v podobě rozhodovacího stromu a způsobu hledání optimální integrační cesty.

Během dalšího projektu (Projekt 4 - Budování ODS (2005 - 2006)), na kterém jsem se podílel, jsem prakticky ověřil principy popsaných rozhodovacích procesů. Zde jsem z hlediska projektového řízení zabýval zejména **fází testování**, jejíž pozorovaná specifika jsem popsal a vysvětlil ve vztahu k integračním projektům v [KARAS, 2006a] a v kapitole 15 Testování.

Konceptuální rámec těchto poznatků tvoří integrační projekt, přesněji řečeno, jeho charakteristika. Optimální životní cyklus projektu (tedy jeho charakteristiku) popisuje projektová metodika, proto bylo mou snahou nalézt metodiku specifickou pro integrační projekty. V praxi jsem se setkal s různými typy metodik, které byly uplatňovány na integrační projekty (viz kapitola 3 Výběr mateřské metodiky), žádná z nich však neodpovídala zcela potřebám integračních projektů. Ve snaze nalézt odpovídající **projektovou metodiku** jsem

analyzoval informační zdroje a pomocí dotazníku jsem provedl i průzkum u komerčních firem (viz kapitola 3 Výběr mateřské metodiky). Tato snaha nebyla úspěšná, proto jsem se rozhodl využít stávající obecnou projektovou metodiku a provést syntézu této metodiky a získaných poznatků ve výslednou metodiku integračních projektů, která je popsána v této práci.

Metodika vznikala paralelně s tím, jak se mi dařilo identifikovat jednotlivé specifické rysy integračních projektů. Její úpravy probíhaly formou iterací podle trvání projektů, z kterých jsem čerpal praktické poznatky. Metodika byla podle poznatků z těchto projektů v kombinaci s nalezenými informacemi z informačních zdrojů zdokonalována. Podle projektových fází obecné metodiky jsem sestavoval fáze specifické metodiky a následně je metodiku ověřoval dle možností v praxi.

Výsledky pozorování a interview z projektů jsou zachyceny v projektových dokumentech (zápisy z jednání, analýzy, technická dokumentace), které ovšem nejsou veřejně přístupné (chráněny obchodním tajemstvím).

Ve snaze provést praktické ověření postupů popsaných v metodice jsem se dle možností snažil uplatnit tyto postupy v integračních projektech, na kterých jsem se podílel. Z praktických důvodů není možné provést komparativní analýzu zvýšení efektivnosti, neboť tento přístup by vyžadoval realizovat tentýž projekt nejprve bez informací a postupů poskytnutých v této práci a podruhé realizovat totéž s nimi. Tento postup není možný vzhledem k finanční náročnosti tohoto typu projektů. Tato analýza však není nutná, neboť lze aplikovat obecnou logickou dedukci, z které vyplývá, že v případě bez užití navržené metodiky by čas k vyřešení daného problému byl sumou času potřebného k nalezení vhodného postupu a času nutného k jeho aplikaci, a v případě užití metodiky bude výsledný čas sumou času nutného pro hledání optimálnějšího postupu (může být roven nule) a času nutného k aplikaci daného postupu. Za objektivního předpokladu, že pro hledání vhodného postupu v obou případech bude užit stejný algoritmus, dojdeme k závěru, že čas pro nalezení nejlepšího postupu za daných podmínek není nikdy v druhém případě větší než čas v prvním případě. Jelikož v obou případech hledáme nejlepší postup, pak čas pro aplikaci postupu je v obou případech shodný. Můžeme tak závěrem konstatovat, že čas v druhém případě je vždy kratší nebo maximálně stejný než čas v prvním případě, a z toho odvodit závěr, že aplikace metodiky za jinak stejných podmínek je efektivnější, přinejhorším stejně efektivní.

Při práci s informačními zdroji došlo k významnému omezení, které bylo způsobeno tím, že v některých částech práce jsem využívat i zdroje, které nejsou veřejně dostupné. Druhým omezením, s kterým jsem vědomě pracoval, bylo pravděpodobné komerční zkreslení některých informačních zdrojů, kterému se však nelze objektivně vyhnout, neboť některé ze zkoumaných nových postupů či technologií nebyly dosud publikovány v žádné, mně známé, objektivní publikaci.

1.5 Struktura práce

Hlavním logickým pojítkem celé práce je životní cyklus projektu popsaný v mateřské metodice, do kterého jsou v logických celcích zasazovány charakteristické prvky integračních projektů.

Práci jsem je rozdělena do následujících logických celků, resp. kapitol:

1 Úvod

- přesné vymezení oblasti zkoumání
- cíle práce a její přínosy
- postup a omezení při zpracovávání práce
- struktura a návaznosti práce
- aktuální stav problematiky ve světě

2 Charakteristika integračních projektů

- základní charakteristika integračních projektů
- typologie integračních projektů

3 Výběr mateřské metodiky

- typologie metodik relevantních k integračním projektům
- průzkum užívaných integračních metodik
- postup při výběru mateřské metodiky

4 Životní cyklus integračního projektu

- východiska z mateřské metodiky
- detailní charakteristika integračního projektu ve vazbě na jeho životní cyklus
- provázanost následujících kapitol na metodiku

5 Dimenze integračního projektu

- zkoumání odlišností integračního projektu v dimenzích mateřské metodice

6 Projektový tým

- struktura týmu integračního projektu

7 Business case

- šablona business case dokumentu
- přínosy, náklady, rizika integračních projektů z pohledu top managementu
- proces rozhodování o integračním projektu

8 Požadavky v integračních projektech

- typologie integračních požadavků

9 Integrační proces a integrační případ

- definice integračního procesu a případu; jejich vztah
- typologie integračních případů
- řešení integračních případů

10 Integrační postupy

- definice termínů integrační koncepce, přístup, technologie, produkt, standard a jejich vzájemný vztah
- popis integračních koncepcí, přístupů a technologií, hledání jejich interních i vzájemných vztahů
- rozhodovací strom pro výběr integrační technologie

11 Model komplexní integrované architektury PIS

- návrh modelu spolupracujících integračních komponent v podnikovém integračním systému

12 Modely integračních nástrojů

- funkcionalita a komunikační schéma hlavních integračních komponent

13 Rozšířený model komplexní integrované architektury PIS

- příklad rozšíření modelu o sdílenou komponentu - BRE

14 Výběr integrační platformy

- proces výběru integrační platformy
- návrh kritérií pro výběr integračního serveru

15 Testování

- popis procesu testování
- charakteristika testování v integračních projektech - obsahové a procesní hledisko

16 Závěr

- shrnutí hlavních přínosů práce

17 Přílohy

- přehled integračních koncepcí, přístupů, technologií a produktů
- stručný popis reálných projektů, které byly užity při sestavování a testování metodiky
- přehled důležitých odborných termínů
- vysvětlení užitých zkratek
- přehled použitých informačních zdrojů

1.6 Aktuální stav ve světě

Tato práce je na pomezí dvou oborových oblastí zkoumání - **projektového řízení** a **informačních technologií**. Z projektového řízení vychází z životního cyklu a charakteristických rysů projektů, v informačních technologiích vychází ze skupiny integračních technologií, resp. postupů.

1.6.1 Projektové řízení

Z pohledu této práce je z projektového řízení podstatná charakteristika projektů informačních systémů, kterou nalézáme především v metodikách pro implementaci informačních systémů. Vedle neveřejných metodik vyvinutých komerčními firmami pro vlastní potřeby existují metodiky veřejné.

Projektové metodiky se rozdělují do dvou základních skupin (dle [CHARVAT, 2003] a [BUCHALCEOVÁ, 2005]). V každé skupině je uveden výčet několika vzorových metodik.

A. Metodiky projektového řízení (globální) - popisují projekty a jejich řízení na rámcové úrovni. Patří sem například:
    - Capability Mature Model for SW (CMM)
    - Object-oriented Process, Environment and Notation (OPEN)
    - Rational Unified Process (RUP) ([BUCHALCEOVÁ, 2005] ji řadí mezi projektové metodiky)
    - Enterprise Unified Process (EUP) ([BUCHALCEOVÁ, 2005] ji řadí mezi projektové metodiky)
    - Projects in Controlled Environments 2 (PRINCE2)
    - System Development Life Cycle (SDLC)
    - Multidimensional Management and Development of Information System (MMDIS)

B. Vývojové (projektové) metodiky - koncentrují se na detailní popis návrhu a vývoje systému. Tyto se dále dělí na

    a) Rigorózní (tradiční, těžké) - představují tradiční způsob vývoje informačního systému vycházející z přesně definovaných procesů, činností a artefaktů. Do této skupiny se zahrnují:
        - Unicycle
        - Code-and-Fix
        - V-model
        - Waterfall model
        - Open source
        - Spiral

- Synchronize and Stabilize
- Reverse Engineering Development
- Structured System Analysis and Design Method (SSADM)
- Pramis
- Offshore Development

b) Agilní (odlehčené, lehké) - mezi charakteristické rysy patří využívání neformální komunikace, méně formálních pravidel a dokumentů. Místo přesně daných procesů definují principy a praktiky. Jsou vhodné pro menší projektové týmy. K těmto metodikám patří:

- Extrémní programování (XP)
- Agilní modelování (vychází z XP)
- Scrum
- Crystal methodology
- Dynamic Systems Development Methodology (DSDM)
- Rapid Application Development (RAD)
- Adaptive Software Development (ASD)
- Lean development
- Feature-Driven Development (FDD)

Pro menší projekty je možno využít pouze některou z vývojových metodik, pro větší projekty se doporučuje kombinace metodiky projektového řízení a vývojové metodiky. Srovnání jednotlivých metodik je k dispozici např. v [CHARVAT, 2003] a [BUCHALCEOVÁ, 2005]. Základní informace o zmíněných metodikách je možné dohledat v informačních zdrojích. Pro tuto práci je podstatné, že žádná z těchto metodik není určena přímo pro integrační projekty. Metodika popisovaná v této práci se v tomto členění řadí mezi **středně těžké vývojové metodiky a je cíleně připravena koexistovat s metodikou projektového řízení MMDIS**, z které vychází. Do skupiny tradičních metodik se řadí zejména z důvodu velkého rozsahu integračních projektů a povahy integračních platforem jako kritických podnikových aplikací. Z konceptuálního hlediska má prezentovaná metodika patrně nejblíže k SSADM, odlišuje se však použitými postupy.

V této oblasti jsou pro tuto práci důležité dva výzkumné projekty - MMDIS a MeFIS.

MMDIS (Multidimensional Management and Development of Information System)

je metodika, jejímž cílem je vývoj a údržba komplexního a integrovaného IS/IT, který optimálně podporuje dosažení podnikových cílů. Tato metodika je vyvíjena na katedře informačních technologií VŠE v Praze pod vedením profesora Voříška od počátku 90. let. Původní název metodiky byl MDIS (Multidimensional Development of Information System), jejíž základní publikací je [VOŘÍŠEK, 1997], postupně byla rozvinuta do metodiky MMDIS, v které byl posílen důraz na řízení informačních systémů. MMDIS je popsána ve [VOŘÍŠEK]. Popisovaná metodika integračních projektů se zabývá primárně vývojem integrovaného systému, proto je pro ni primárním zdrojem MDIS, ovšem se všemi úpravami MMDISu ve vývojové části metodiky (proto je v textu jednotně používán odkaz na MMDIS, nikoli MDIS).

Jelikož z MMDISu vychází popisovaná metodika integračních projektů, jsou vzájemnému vztahu těchto metodik věnovány dvě samostatné kapitoly - 4 Životní cyklus integračního projektu a 5 Dimenze integračního projektu.

MeFIS (Methodical Framework for Information Systems Development)

MeFIS poskytuje základ pro vývoj a kategorizaci metodik informačních systémů. Byl vytvořen rovněž na katedře informačních technologií VŠE v Praze, základní publikací MeFISu je [BUCHALCEOVÁ, 2005]. Kategorizaci metodiky integračních projektů prezentované v této práci definuje dle navržené struktury MeFISu tabulka 1 - Popis metodiky dle MeFIS (zdroj: autor).

Položka	Význam
ID metodiky	MIP
Název metodiky	Metodika integračních projektů (Methodology of Integration Projects)
Autoři	Ing. Jan Karas
Rok vzniku	2006
Globálnost	Projektová metodika
Rozsah	Fáze - řeší částečně IST (skrze integrační strategii), UST, GAN, DAN, IMP, ZAV, PUR; neřeší GST. Dimenze - řeší HW (orientačně), SW, DAT, FUN, PRA (orientačně), ORG, EKO; neřeší uživatelské rozhraní (UI). Role - definuje 27 rolí v 6 týmech kombinovaných ze strany zákazníka, integrátora a dodavatele integrační platformy.
Váha metodiky	MM (středně těžká) - definuje základní procesní návaznosti, artefakty a činnosti v projektu a zároveň principy a praktiky v jeho fázích.
Typ vývoje	Řeší NEW (vývoj nového řešení), INT (integrace řešení), UPG (rozvoj a rozšíření řešení); neřeší TYP (implementace typového řešení).
Doména	EAI
Přístup k řešení	Nelze obecně říci vzhledem k portfoliu integračních technologií, ale hlavně ST - strukturovaný vývoj, OR - objektový vývoj s relační databází.

tabulka 1 - Popis metodiky dle MeFIS (zdroj: autor)

Poznámka k váze metodiky: Váha metodiky je součinem velikosti a hustoty metodiky. Posuzuje se podle podrobnosti, přesnosti, relevance, tolerance a měřítka metodiky. [BUCHALCEOVÁ, 2005] neuvádí deterministickou metodu určení váhy metodiky, proto jsem se řídil charakteristikou těžkých a lehkých metodik a srovnáním s kategorizací metodik uvedených v [BUCHALCEOVÁ, 2005].

[BUCHALCEOVÁ, 2005] uvádí pomocnou tabulku pro určení optimální váhy metodiky v závislosti na dvou atributech projektu - počtu lidí v týmu (A07) a důležitosti systému (A05). Z tabulky 26 - Atributy zkoumané skupiny integračních projektů podle MeFIS (zdroj: autor) plyne, že integrační projekty spadají přibližně do kategorie H20 - H50, které jsou na pomezí mezi lehkými a těžkými metodikami a středně těžká metodika se pro ně jeví jako nejvhodnější.

Metodika pro integrační projekty přispívá k plnění jednoho z hlavních cílů MeFISu, a to vytvořit rodinu projektových metodik podle domén informačních technologií zastřešených metodikou MMDIS - v tomto případě pokrývá doménu EAI. Odborným termínem pro rodinu metodik je metodický rámec, který definuje [BUCHALCEOVÁ, 2005] jako "kolekci metodických vzorů pro různé domény, typy řešení a způsoby řešení spolu s principy a procesy pro vytvoření konkrétní metodiky."

V [CHLAPEK, 2005] jsou prezentovány principy řízení a koordinace projektů v MeFISu. Odděluje se zde řídící a věcný postup projektů. Zatímco řídící postup má 4 fáze - přípravu, naplánování, provedení a ukončení projektu, věcné postupy představují dle [CHLAPEK, 2005] "rozklad prací členěný do etap a činností konkretizovaných pro věcný typ projektu". Tato práce se primárně zaměřuje na věcný postup v integračních projektech.

1.6.2 Informační technologie

Oblast integračních technologií je podskupinou informačních technologií. Jak konstatuje [ARRIBA], "pole integrace podnikových aplikací je v podstatě řízeno trhem a je takřka exklusivně vyvíjeno softwarovými dodavateli a konzultačními společnostmi (včetně velké pětky). Akademická účast je výrazně omezena, přestože se objevují potenciální spojení s komunitami zabývajícími se výzkumným re-inženýringem. Hlavní směr EAI je

výrazně (webově) technologicky orientovaný a publikace vztahující se k EAI jsou taktéž technologické spíše než akademické."

Existuje řada informačních zdrojů včetně publikací, které se zabývají integračními technologiemi. Přehledovými publikacemi jsou např. [LINTHICUM, 2003], [LINTHICUM, 1999], [BUSSLER, 2003]. Další publikace se specializují jen na určitou podmnožinu těchto technologií, jako např. [MORAGENTHAL, 2005a] (EII), [SHARMA, 2001] (J2EE), [KIMBALL, 1998] (DWH), , [CHAPPELL, 2004] (ESB). Početnou skupinu publikací tvoří skupina technologií souvisejících se SOA a webovými službami - např. [KAYE, 2003], [BARRY, 2003], [CERAMI, 2002] a [NEWCOMER, 2002]. Integrační komponenty jsou významnou součástí architektury podnikových systémů, o které se píše např. v [DOHNAL, 1997], [BASS, 2003], [HOHMANN, 2003], [SESSIONS, 2003], [MARCUS, 2002] a [ALBIN, 2003]. Integrace aplikací souvisí i s vnitřní softwarovou architekturou aplikací, které se věnují [FOWLER, 2002] a [ARLOW, 2003].

V akademické sféře se s integračními technologiemi setkáváme v několika směrech. Zaprvé jsou to výzkumné projekty zaměřené na jednotlivé integrační technologie ve snaze rozšiřovat jejich možnosti - sem patří např. práce týkající se business pravidel ([BAJEC, 2005b], [BAJEC, 2005a]), metadat ([KASHYAP, 2000]), SOA, SOE a sémantických webových služby ([PAPAZOGLOU, 2005], [HALL, 2005], [SVÁTEK, 2004], [ARPINAR, 2005]). Druhým směrem, se kterým se v informačních zdrojích setkáváme, je snaha nastavit řád v chaosu integračních postupů, které produkuje komerční sféra, a připravit tak zázemí pro další výzkumné projekty. Z tohoto směru je řada přednášek z konferencí. Současně jsou tyto informace prezentovány i formou podkladů k přednáškám na univerzitách, kde jsou EAI věnovány samostatné předměty. Jako příklady informačních zdrojů uvádím [TSE], [PEPER], [LAND, 2004] a [BOND, 2001]. Třetím směrem jsou praktické projekty implementace integračních technologií na vědeckých pracovištích a studie vybraných aspektů integračních technologií v tržním prostředí, zpravidla ve formě diplomových a disertačních prací. Jejich součástí je většinou i analýza dostupných integračních technologií ve vztahu k danému projektu. Mezi tyto informační zdroje patří např. [SKOGLUND, 2002], [KRAUS, 2002] a [STRÜVER, 2002].

Jádrem této práce ovšem není zmapovat nebo nalézat nové integrační technologie, ale s jejich znalostí navrhnout efektivní postupy pro jejich implementaci ve formě metodiky. Proto jsou informační zdroje z oblasti integračních technologií v této práci používány pouze pro získání přehledu o šíři této oblasti a pro navázání se na tuto oblast. Z toho důvodu se v této práci ani nesnažím analyzovat jednotlivé technologie do hloubky, neboť to není relevantní k cíli této práce.

Mapování současného stavu ve světě integračních technologií se věnuje kapitola 10 Integrační postupy (a související přílohy) a tato práce na tento stav navazuje. Kritické srovnání s dostupnými informačními zdroji je popsáno v kapitole 10.1 Přehled integračních postupů v informačních zdrojích.

1.6.3 Integrace aplikací

Blíže k této práci jsou informační zdroje, které se zabývají fází návrhu a definují vzory pro řešení integračních úloh na bázi zasílání zpráv - např. [HOHPE, 2003b] a [HOHPE]. Tyto vzory jsou plně kompatibilní s prezentovanou metodikou, neboť na detailní úrovni rozpracovávají řešení integračních případů v konkrétní technologii.

[HALLE, 2002] a [ROSS, 2003] popisují tzv. Business Rules Approach, což je postup, jakým transformovat požadavky na systém do business pravidel, která lze implementovat do informačního systému. Pro jejich implementaci se využívají produkty typu BRE, jejichž vztah k integračním komponentám řeším v kapitole 13.2 BRE v integračním projektu. Samotný postup je nezávislý na integračních technologiích a lze jej využít i jako podpůrný postup v integračních projektech při implementaci BRE. Této oblasti se věnují na Univerzitě v

Ljubljaně na fakultě Computer & Information Science (viz [BAJEC, 2005b] a [BAJEC, 2005a]).

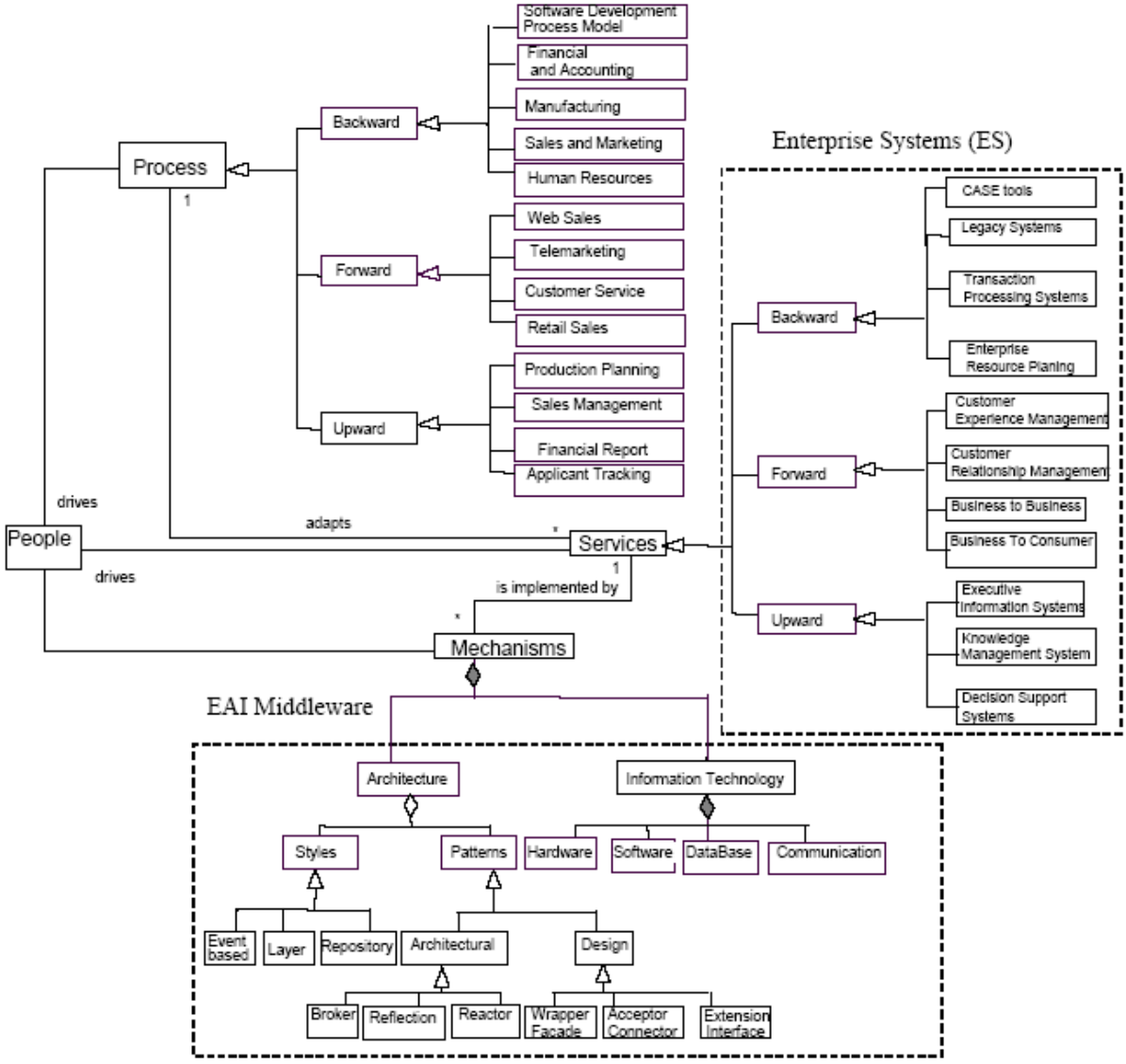
V informačních zdrojích nalézáme několik modelů, jejichž smyslem je poskytnout rámec pro problematiku integrace aplikací. Přehled nejznámějších nabízí tabulka 2 - Přehled rámců pro EAI v informačních zdrojích (zdroj: autor).

Název	**Popis**	**Zdroj**
MDA pro EAI	Popisuje 5 modelů: o Technologicky specifický model - definuje základní technologickou platformu pro fyzické propojení a spolupráci aplikačních systémů. o Transakčně servisní model - definuje transakční podporu pro aplikační služby. o Generický aplikačně servisní model - definuje služby požadované pro automatizaci podnikových procesů. o Intra aplikační model - definuje funkcionalitu a workflow, které musí být poskytováno aplikačně servisním modelem. o Inter aplikační model - definuje informační toky a vztahy s vnějším okolím podniku.	[MOSAWI, 2006]
EAIF (EAI Framework)	Kombinuje tři EAI pohledy - integraci interních procesů, externích integraci procesů a integraci rozhodovacích procesů s třemi úrovněmi Brownova konceptuálního modelu integrace - procesní, služeb a mechanismů (viz obrázek 1 - Rozšířený EAI framework (zdroj: [LOSAVIO, 2005])).	[LOSAVIO, 2005], [LOSAVIO, 2002], [LOSAVIO, 2003]
Zachman framework	Je zaměřen obecně na vývoj IS. Kombinují se dvě dimenze - role (plánovač, vlastník, návrhář, vývojář, subkontraktor) s elementy projektu (data, procesy, síť, role, časování, motivace).	[LOSAVIO, 2005]
Whitten framework	Je zaměřen obecně na vývoj IS. Kombinuje vzájemně role (projektový manažer, systémový analytik, vlastník systému, systémový uživatel, systémový designer, systémový vývojář), podnikové (zlepšování znalostí, podnikových služeb a procesů, komunikace a spolupráce) a technologické tahouny (databázové, softwarové technologie a technologie rozhraní).	[LOSAVIO, 2005]
Zoran framework	Integruje komponentový vývoj (CBD - Component - Based Development) s referenčním modelem ISO pro distribuované zpracování (RM-ODP - Reference Model of Open Distributed Processing). Výsledkem jsou tři modely: o Model podnikové architektury (EAM - Enterprise Architecture Model) - popisuje chování systému v kontextu podniku o Model architektury systému (SAM - System Architecture Model) - definuje systém pomocí služeb a jejich interakcí o Model architektury distribuce (DAM - Distribution Architecture Model) - popisuje infrastrukturu formou distribuovaných služeb	[LOSAVIO, 2005]
Cummins	Definuje Enterprise Integration Architecture (EIA), která	[LOSAVIO, 2005]

framework	je charakterizována dle [LOSAVIO, 2005] "distribuovanými výpočty, komponentovými aplikacemi, událostmi řízenými procesy, volně vázanými podnikovými funkcemi, informacemi podporujícími rozhodování, řízením workflow, internetovým přístupem, personalizací rozhraní".	
Business Engineering framework	Obsahuje rozšířený referenční model podnikového inženýrství pro integrační účely. Tento model je popsán v kap. 9.5 Modelování procesů.	[SCHELP, 2005]
Information System Architectural framework for EAI	Zavádí do obecného modelu (v UML) architektury IS stereotyp "IT integrační blok" a "IT integrační službu", pomocí kterých lze v modelu zachytit integrační vazby.	[VASCONCELOS, 2004]

Poznámka k uvedeným zdrojům: [LOSAVIO, 2005] se zabývá detailním porovnáním těchto konceptuálních modelů, proto odkazuji na tento zdroj, přes který je možné dohledat primární zdroje.

tabulka 2 - Přehled rámců pro EAI v informačních zdrojích (zdroj: autor)



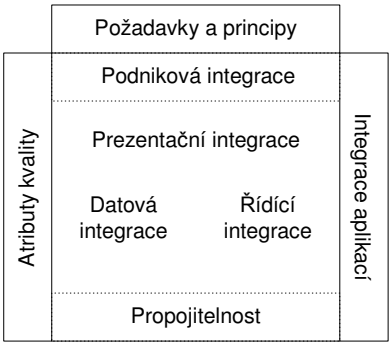
obrázek 1 - Rozšířený EAI framework (zdroj: [LOSAVIO, 2005])

Tyto modely zachycují oblast integrace aplikací z jiného úhlu než tato práce. Zde je na problematiku nahlíženo skrze integrační projekt, jeho fáze a dimenze, zatímco zmíněné modely jsou pohledem architekta podnikového systému. Pohled architekta podnikového systému je širší, neboť vedle integrační platformy, na kterou se tato práce soustředí, zvažuje i jiné možnosti řešení úloh kladených na podnikový systém - prostřednictvím podnikových procesů a aplikací. Chápejme proto tyto modely jako alternativní, jinak cílený pohled na stejnou oblast zájmu.

1.6.4 Podniková integrace

V informačních zdrojích se vedle termínu integrace podnikových aplikací setkáváme rovněž s příbuzným termínem podniková integrace (EI - Enterprise Integration). [SMITH, 2002] ji definuje takto: "Cílem podnikové integrace je zajistit včasnou a přesnou výměnu konzistentních informací mezi podnikovými funkcemi pro podporu strategických a taktických podnikových cílů tak, aby se jevila jako nepřerušená." Obsah podnikové integrace popisuje obrázek 2 - Model podnikové integrace (zdroj: [SMITH, 2002]), který znázorňuje její hlavní technické komponenty (vedle nich jsou za důležité považovány rovněž manažerské a organizační aspekty):

- Požadavky a principy - zabývá se podnikovými stimuly a řídící principy, které pomáhají při vývoji podnikové architektury.
- Podniková integrace se zabývá návrhem a modelováním podnikových procesů (BPR).
- Prezentační integrace - zabývá se integrací podnikových znalostí a podnikových procesů pro sdílení aplikací, služeb a klíčových kompetencí mezi zaměstnanci, zákazníky, partnery a dodavateli.
- Datová integrace - zabývá se integrací podnikových dat.
- Řídící integrace - zabývá se různými typy zasílání zpráv mezi aplikacemi s ohledem na různé typy komunikace a protokolů.
- Propojitelnost - se vztahuje k propojení workflow, dat a služeb a řeší propojení mezi aplikacemi.
- Atributy kvality - zabývá se kvalitativními charakteristikami, jako např. výkonnost, závislost a bezpečnost.
- Integrace aplikací - se zbývá tím, aby aplikace pracovali společně s využitím různých mechanismů, jak je např. automatická notifikace o událostech.



obrázek 2 - Model podnikové integrace (zdroj: [SMITH, 2002])

Pojetí podnikové integrace se tak blíží pojetí integrace informačních systémů, jak ji pojímá [VOŘÍŠEK, 1997], viz obrázek 6 - Úrovně integrace informačního systému (zdroj: [VOŘÍŠEK, 1997]).

Z vymezeného obsahu podnikové integrace plyne, že námi zkoumaná integrace podnikových aplikací tvoří její podstatnou část - jmenovitě integraci aplikací a podstatnou měrou zasahuje i do zbývajících typů integrace.

Po představení dvou oborů - projektového řízení a informačních (resp. integračních) technologií, které poskytují východiska pro tuto práci, se nyní podívejme na výzkumné projekty, které se stejně jako tato práce zabývají metodikou pro integrační projekty a definujme jejich pozici k této práci.

1.6.5 Výzkumné projekty

Projekt GBOU

Záměrem projektu GBOU je poskytnout metodiku a související nástroje pro podporu integrace nesourodých podnikových aplikací, které nemusely být navrženy pro vzájemnou koexistenci. Tento projekt běží od října 2002 a zastřešuje jej konzorcium ARRIBA (Architectural Resources for the Restructuring and Integration of Business Applications), jehož členy jsou evropské univerzity a řada společností z praxe. Jak se uvádí v základním dokumentu projektu [ARRIBA], práce na projektu jsou rozděleny do několika pracovních skupin - Vrije Universiteit Brussel - Programming Technology Lab (PROG), Universiteit Gent - Information Technology (INTEC), Universiteit Antwerpen - Lab On Reengineering (LORE) a Concurrency, Université Catholique de Louvain - Département d'Ingénierie Informatique (INGI) a Universität Bern - Software Composition Group (SCG).

Výstupy projektu jsou rozděleny do tří skupin - pro veřejnost, pro partnery a pro členy konzorcia. V publikacích pro veřejnost není informační zdroj, který by prezentoval ucelenou metodiku pro integrační projekty. Nalézáme zde spíše technologické návrhy, jak řešit specifické integrační úlohy. Lze tedy jen předpokládat, že postupně vznikne rámec jakési metodiky, který spojí tyto příspěvky do kompaktního celku.

Projekt GIF (Global Integration Framework)

Tento projekt nastartoval v roce 2004 pod hlavičkou konzorcia Integration Consortium (viz [Integration Consortium]). Jeho záměrem je spojit dohromady průmyslové standardy, best practices v oblasti integrace, formální modelovací techniky a další iniciativy pro vytvoření normativního přístupu k systémové integraci. Jeho členy jsou převážně komerční firmy. Konkrétně se soustředí na tyto prvky (dle [SCHMIDT, 2004c]):

- univerzální slovník integrační terminologie,
- referenční architektura nezávislá na dodavateli,
- integrační metodika obsahující normativní šablony a standardy,
- centrální registr rozhraní kompatibilních s GIF,
- softwarový framework (možná open source) integračních služeb,
- integrační datový model a související nástroje,
- výukový program,
- certifikační proces pro jednotlivce i organizace.

Ve vztahu k této práci se tento projekt překrývá v prvních třech bodech - slovníku, referenční architektuře a metodice. Jeho výsledky jsou očekávány v roce 2007. Výsledky tohoto projektu bude dle dostupných informací publikovány pro jedince a univerzity zdarma, pro komerční sektor za poplatek. V jsou dokumenty sdíleny pouze v rámci konzorcia.

10 základních principů metodiky je popsáno v [SCHMIDT, 2003b], jsou to tyto:

- Soulad EAI plánů s podnikovou strategií.
- Nejdříve konsolidovat, až poté integrovat.
- Používat procesní přístup pro dosažení end-to-end řešení.
- Nastavit jasné hranice vlastnictví a odpovědnosti.
- Prosazení EAI architektury.
- Nařizovat integrační požadavky pro nové aplikace.
- Vyvíjet společnou prezentaci dat a procesů.
- Testovat brzy a často.
- Neustále zdokonalovat rozhraní, aby se nikdy nestala zastaralými.

- Vyvíjet podnikové procesy prostřednictvím experimentování.

Metodika pro integrační projekty prezentovaná v této práci je s těmito principy v souladu.

Pod hlavičkou EAI Industry Consortium, které je předchůdce Integration Consortium, postupně vzniká metodika Total Business Integration (TBI) která je určena pro integrační projekty a je rovněž nezávislá na konkrétní technologii. Má definovány 4 projektové fáze:

- Specifikace - rozpadá se na definici projektu, analýzu podnikových procesů, analýzu technických požadavků a plánování zajištění kvality.
- Návrh - obsahuje logický návrh a architekturu.
- Vývoj - zahrnuje návrh integrace, programování a testování.
- Zavedení - představuje akceptační testování a provozování.

K jednotlivým fázím má definované výstupy a pro tyto výstupy jsou připraveny šablony.

Tato metodika má prakticky shodné zaměření jako metodika popisovaná v této práci. Podle dostupných informací však její vývoj nebyl dosud finalizován a ve veřejně dostupné verzi [EAI INDUSTRY CONSORTIUM, 2004] obsahuje pouze základní rámec celé metodiky. Metodika TBI je v aktuální verzi dostupná pouze členům konzorcia. Vedle zaměření metodiky TBI nalézáme další shodný rys v opření se této metodiky o globální metodiku, The Open Group Architecture Framework (TOGAF, [THE OPEN GROUP, 2003]), podobně jako se tato metodika opírá o globální metodiku MMDIS. Vzájemná vazba mezi TBI a TOGAF je mapována v [BLEVINS, 2004].

Projekt OpenEAI (Open Source Enterprise Application Integration Software and Methodology)

Cíle projektu jsou definovány takto:

- Metodika pro analýzu a definování integrace.
- Protokol pro zasílání zpráv na bázi XML, který poskytne specifikaci pro podnikové zprávy a obecné chování pro aplikace, které tyto zprávy zpracovávají.
- Balík základních API postavených na standardech, která poskytnou základní bloky pro integraci.

Zakladatelem tohoto projektu je univerzita v Illinois v Chicagu (v březnu 2001), která by měla být i prostředím pro referenční implementace. Na domovské stránce tohoto projektu [OpenEAI] jsou zpřístupněné veřejné informační zdroje, z kterých jsou po tuto práci relevantní [JACKSON, 2005] a [JACKSON, 2003].

Metodika popsaná v [JACKSON, 2005] popisuje tyto projektové fáze:

1. Analýza (dokumentace požadavků, identifikace nových podnikových objektů, specifikace stávajících podnikových objektů, definice struktury nových objektů, specifikace zasílání zpráv mezi objekty).
2. Definice zpráv.
3. Generování zpráv jako objektů v Javě a souvisejících artefaktů.
4. Vývoj, dokumentace a testování aplikací pro zasílání zpráv.
5. Úprava podnikové dokumentace artefaktů.
6. Instalace do produkce.
7. Provoz.

Dále tento dokument detailně popisuje jednotlivé fáze včetně šablon a konkrétních příkladů z prostředí univerzity. [JACKSON, 2003] popisuje integrační scénáře pro zasílání zpráv v závislosti na použité technologii.

Shrnu-li vztah tohoto projektu k této práci, tak konstatuji, že projekt je úzce zaměřený na integrační přístup zasílání zpráv a Javu. Vzhledem k plánované referenční implementaci na univerzitě je hodně technologicky zaměřený. Popis metodiky není dostatečně detailní,

aby bylo možné provést srovnání navržených postupů, neboť se soustředí primárně na jednotlivé prvky řešení a použité technologie spíše než na použité projektové postupy.

EIM (Enterprise Integration Methodology)

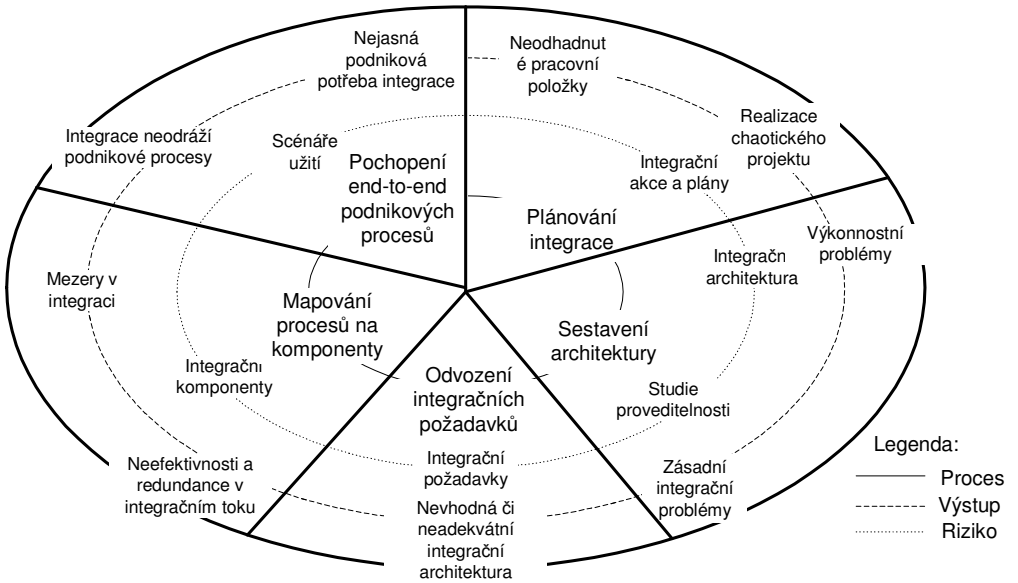
EIM byla publikována v [LAM, 2004] v roce 2004 pod záštitou IEEE Computer Society autory Wingem Lamem (Universitas 21 Global) a Venky Shankararamanem (National University of Singapore). Životní cyklus integračního projektu v EIM se skládá z 5 fází:

1. Pochopení end-to-end podnikových procesů - cílem je definovat, jak by měly fungovat interní i externí procesy bez ohledu na IT, provádí se tedy BPR. Tato fáze je považována za klíčovou pro úspěch projektu.
2. Mapování procesů na komponenty - specifikuje se, jak podnikové aplikace a integrační komponenty budou podporovat nadefinované procesy. Plánované změny oproti stávajícímu stavu jsou rozdělovány do menších projektů.
3. Odvození integračních požadavků - nutné změny jsou rozděleny do tří skupin podle dopadu na:
    a. Legacy systémy, které je potřeba integrovat.
    b. Manuální procesy, pro jejichž automatizaci je potřeba implementovat nové IT systémy.
    c. Manuální procesy, pro jejichž automatizaci je potřeba rozšířit stávající legacy systémy.

    Pro integrační požadavky je navrženo 10 kategorií - propustnost rozhraní, doba odezvy, objem dat, naléhavost (online / se zpožděním), datový formát, protokol přenosu, protokol integrační infrastruktury, odolnost proti chybě, perioda přenosu, zabezpečení.

    Tyto požadavky jsou podkladem pro výběr integračního produktu.
4. Sestavení architektury - definuje výběr a vzájemnou provázanost integračních komponent, vyžaduje vysokou znalost v oblasti integračních technologií.
5. Plánování implementace integrované architektury - sestavují se navazující projektové plány pro jednotlivé integrační subprojekty. V této fázi se subprojekty realizují tradičními fázemi - návrhem, implementací, testováním a nasazením do provozu.

Každá z těchto fází má definovaný proces, výstupy a rizika, jak ukazuje obrázek 3 - Enterprise Integration Methodology (zdroj: [LAM, 2004]).



obrázek 3 - Enterprise Integration Methodology (zdroj: [LAM, 2004])

V porovnání s touto prací se EIM odlišuje tím, že v úvodní fázi zahrnuje i BPR a v následných fázích zahrnuje i úpravy stávajících a implementaci nových podnikových aplikací. Uvedené kategorie integračních požadavků odpovídají spíše charakteristikám integračních případů, než globálním integračním požadavkům. Vzájemný vztah EIM a metodiky v této práci se shoduje se vztahem, který je diskutován u následující metodiky pro vývoj integrovaných IT infrastruktur.

Metodika pro vývoj integrovaných IT infrastruktur (Methodology for the Development of Integrated IT Infrastructures)

Tato metodika byla prezentována v [THEMISTOCLEOUS, 2006] na konferenci v roce 2006, jejími autory jsou Marinos Themistocleous a Zahir Irani ze School of Information Systems, Computing and Mathematics z Brunelské univerzity (Uxbridge), ze skupiny Information Systems Evaluation and Integration Network Group (ISEing).

Vzhledem k tomu, že byla publikována až v roce 2006, tedy až v době, kdy byla tato práce dokončována, nebylo možné ji využít jako primární zdroj, ale naopak ke komparaci dosažených závěrů. Zmíněná metodika se potkává s touto prací v několika oblastech - ve vymezení EAI, vztahu projektových metodik vývoje obecných aplikací k integračním projektům, charakteristice integračních projektů, postupem výzkumu a životním cyklem projektu.

Oblast EAI definuje jako "strukturovanou aplikaci technologií, nástrojů, metod, přístupů a plánů, jejímž účelem je vytvořit end-to-end integrované podnikové procesy běžící na IT infrastruktuře", což plně odpovídá pojetí EAI v této práci.

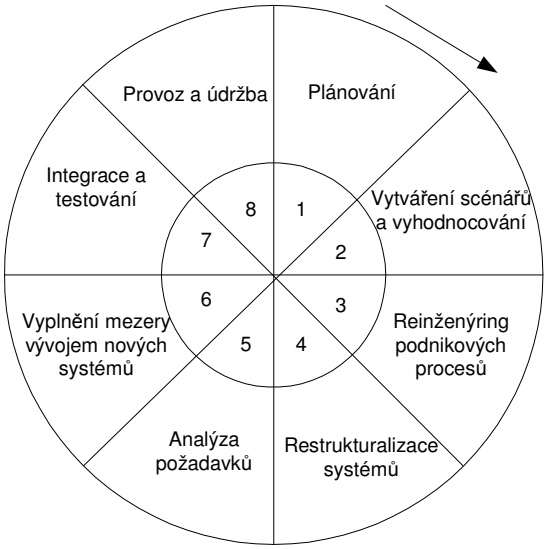
Dále dochází k závěru, že tradiční metodiky pro vývoj obecných aplikací nesplňují požadavky integračních projektů, proto je zapotřebí specifické metodiky. Zároveň konstatuje nedostatek zavedených metodik pro EAI. V této práci se nezávisle provedenou analýzou došlo ke shodným závěrům.

Hlavní rozdíly EAI projektů oproti projektům vývoje obecných aplikací jsou spatřovány v integraci procesů a v tom, že smyslem integračních projektů není vytvoření nové aplikace v PIS, ale propojení aplikací. Tato práce přistupuje k charakteristice projektů na podstatně detailnější úrovni, která pokrývá i tyto dva podstatné rozdíly.

Metodika navazuje na EIM, oproti které zohledňuje i potřebu implementace nových podnikových aplikací a změn stávajících v důsledku změny podnikových procesů a definuje více projektových fází (viz obrázek 4 - Metodika pro vývoj integrovaných IT infrastruktur (zdroj: [THEMISTOCLEOUS, 2006])):

1. Plánování - při plánování integračního projektu se zohledňuje 10 faktorů - náklady, přínosy, překážky, interní a externí tlaky, podpora, IT infrastruktura, IT náročnost, vyhodnocení frameworku pro předpokládané integrační technologie a EAI produkty (převzato z [THEMISTOCLEOUS, 2003]). Tyto faktory jsou vyhodnocovány ve studii proveditelnosti.
2. Vytváření scénářů a vyhodnocování - pro zkoumání variant jsou možné 2 přístupy - oportunistický řeší integrační problémy specifického procesu nebo oddělení, zatímco strategický vyžaduje end-to-end implementaci EAI (viz také [NEWCOMER, 2004]).
3. Reinženýring podnikových procesů - zde se setkává BPR s EAI. Zatímco balíčková řešení typu ERP poskytují obecně použitelné podnikové procesy, EAI přizpůsobuje podnikové procesy konkrétním potřebám podniku.
4. Restrukturalizace systémů - probíhá mapování podnikových procesů na aplikace, odstranění nepotřebných součástí (např. redundatních dat a rozhraní), řeší se migrace dat do cílových systémů bez omezení výkonnosti podniku.
5. Analýza požadavků - zformulování požadavků na změny, které vyplývají z předchozích 3 fází. Zároveň se vyhodnocují vhodné integrační technologie a nástroje.

6. Vyplnění mezery vývojem nových systémů - podnik analyzuje, navrhuje, implementuje a testuje nové systémy, kterými zaplňuje nalezené mezery v podpoře podnikových procesů. K tomu se využívají metodiky pro vývoj obecných systémů.
7. Integrace a testování - implementuje se integrační infrastruktura, integrují se staré i nové systémy, probíhá end-to-end testování.
8. Provoz a údržba - probíhá provoz a rozvoj integrovaného systému podnikových aplikací.



obrázek 4 - Metodika pro vývoj integrovaných IT infrastruktur (zdroj: [THEMISTOCLEOUS, 2006])

Popsaná metodika svým zaměřením pokrývá širší oblast na podstatně nižší úrovni detailu než metodika v této práci. Metodika prezentovaná v této práci se soustředí na řešení integračních úloh, které jsou definovány nadřízenými projekty. Tím může být BPR projekt, řízený svou vlastní specifickou metodikou, nebo jakýkoli jiný projekt, který vyžaduje integraci mezi podnikovými aplikacemi. Zatímco výše uvedená metodika kombinuje tři oblasti - BPR, EAI a vývoj obecných aplikací a zdůrazňuje jejich vzájemné závislosti. Ve výsledku se tedy obě metodiky vzájemně doplňují a jsou kompatibilní, neboť jsou postaveny na společném principu podnikových procesů, potřebě integrovat podnikové aplikace a jejich projektové fáze na sebe navazují.

Autoři metodiky zvolili obdobný vědecký postup, jaký byl použit i v této práci. Vedle použití shodných informačních zdrojů zkoumali formou interview 12 reálných projektů. Z posbíraných údajů identifikovali tato doporučení:

- Klást důraz na plánování před implementací EAI.
- Ověřit funkcionalitu (i opakovaným) pilotním provozem.
- Kritická je vysoká znalost v oblasti integračních IT.
- Výsledky fáze BPR mají zásadní dopad na další fáze.

Při zpracování této práce tvořily informace z reálných projektů rovněž důležitou součást ověřování metodiky. Tyto informace byly sbírány průběžně během projektů na detailnější úrovni.

EAI metodika M. B. Jurica

Tato metodika byla publikována v [JURIC, 2001] a rozděluje životní cyklus integračního projektu na 8 technických aktivit - sběr požadavků, analýzu existujících aplikací, výběr integrační infrastruktury, analýzu problémové domény, návrh, implementaci, testování a nasazení. K těmto fázím popisuje 3 podpůrné aktivity - projektové řízení, konfigurační a

změnový management a komunikaci s prostředím (viz také [CAVELTI, 2004]). Průřezově k těmto aktivitám definuje 4 fáze - úroveň datovou, aplikačních rozhraní, business metod a prezentační.

Ve vztahu k této práci aktivity přibližně odpovídají projektovým fázím a fáze - úrovně základnímu členění integračních přístupů dle typu rozhraní. Metodika M. B. Jurica se přímo neváže na podnikové procesy a naopak má úzkou vazbu na skupinu technologií J2EE na úkor dalších technologií.

1.6.6 Inovativní přístup k problematice

Zvolený přístup ke zpracovávané problematice je inovativní v několika směrech:

- V informačních zdrojích se setkáváme se snahou charakterizovat projekty, včetně integračních projektů. Tato charakteristika má však primárně informativní charakter (viz kapitola 2.1 Obecná charakteristika integračních projektů). Tato práce však nachází souvislosti mezi charakteristikou integračních projektů a jejich metodikou (viz kapitola 16 Závěr).
- Metodiky jsou zpravidla zpracovávány osamoceně pro řešení jedné úlohy (ať už obecné nebo specifické). Tato práce je úzce navázána na metodický rámec MeFIS, který vychází z metodiky MMDIS, a jehož cílem je vytvořit kompaktní sadu metodik. Oproti jiným metodikám využívá zde prezentovaná metodika princip dědění z mateřské metodiky a koncentruje se primárně na odlišné rysy integrační domény.
- Postoj většiny metodik k integračním postupům je dvojí - buďto jsou formulovány natolik obecně, že se jimi vůbec nezabývají, nebo pokrývají pouze omezenou skupinu integračních postupů (typicky např. zasílání zpráv, SOA, tradiční EAI nebo jen datovou integraci). V této práci je prezentován průřezový pohled skrze většinové portfolio integračních postupů, včetně prezentačních, což umožnilo analyzovat vzájemné vztahy integračních postupů, sestavit ucelený model integrace PIS a rozhodovací strom integračních postupů, tedy především pochopit souvislosti ve fázi návrhu. Práce tak představuje integraci aplikací jako silně heterogenní svět s množstvím závislostí.
- Vedle těchto dominantních směrů existuje celá řada sekundárních, z nichž zmíním např. dimenzionální pohled na integrační projekt (vyplývající z metodiky MMDIS), integrační strategii, integrační procesy a případy a jejich vztah k integračním postupům, detailní zkoumání testovacího procesu v integračním projektu, atd.

2. Charakteristika integračních projektů

Tato kapitola představuje prostředí integračních projektů, jejich typické problémy vyplývající z podstaty integrace a ze specifických nároků těchto projektů na projektové řízení. Dále definuje základní typologii integračních projektů dle klíčových hledisek a zasazuje integraci aplikací do kontextu integračních úrovní systémové integrace.

2.1 Obecná charakteristika integračních projektů

Současné integrační projekty jsou postaveny na základech koncepce EAI, která charakterizuje projekty následujícími rysy:

1. Využití stávajících systémů - při přechodu k integrovanému podnikovému informačnímu systému není nutno nahrazovat stávající systémy novými. U starších aplikací stačí většinou provést mírnou modifikaci a připravit rozhraní aplikace.
2. Zlepšení podnikových procesů - integrace umožňuje plynulé navázání dosud roztříštěných procesů a jejich automatizaci, která pak dále implikuje další přínosy. Právě rozhraní oddělených procesů jsou v BPR (Business Process Reengenering) považována za velmi problematickou část podnikových procesů. Možnosti integrace (resp. automatizace) procesů mají často zásadní dopad na stávající podnikové procesy a vyžadují si jejich podstatné redefinování, aby se docílilo maximálních přínosů.
3. Orientace na podnikové procesy - nastavení podnikových procesů je nedílnou součástí konceptu EAI, business analytik začíná vždy svou práci nejdříve zkoumáním podnikových procesů. Toto je jedna ze zásadních odlišností oproti dříve prosazované integraci pouze na úrovni aplikačního middleware.
4. Architektura aplikačních konektorů - ke každé aplikaci je vytvořen tzv. univerzální adaptér, který definuje rozhraní aplikace k ostatním aplikacím. Modernější a robustnější podnikové aplikace dodávané jako ASW mají tyto adaptéry většinou již připraveny. V novější koncepci SOA jsou adaptéry s původně proprietárními rozhraními nahrazeny standardizovanými rozhraními.
5. Předpřipravená integrační platforma - integrační produkty nabízejí širokou škálu technologií pro rychlou a robustní integraci aplikací včetně sdílených komponent, které umožňují efektivní vývoj a provozování platformy.
6. Distribuované prostředí - EAI produkty podporují distribuované heterogenní prostředí bez ohledu na to, zda se jedná o propojení dvou aplikací nebo řádově několika desítek aplikací ve velkém podniku.
7. Rozdíl mezi vnitropodnikovou a vněpodnikovou integrací přetrvává jen v určitých technologiích, z hlediska koncepčního prakticky mizí.
8. Výrazně se projevují požadavky na standardizaci a univerzálnost řešení. Ať už v oblasti aplikačních rozhraní ve formě webových služeb, přes přenášená data ve formě zpráv v kanonickém formátu a snaze unifikovat strukturu a rozhraní ODS, až po standardizaci v oblasti návrhu interní logiky procesů ve formě vzorů.

V praxi jsem identifikoval specifické problémy, s kterými se v integračních projektech často setkáváme:

1. Heterogenní aplikační prostředí - A2A integrace často bez nadsázky znamená propojení dvou či více (u větších projektů řádově několika desítek) výrazně odlišných světů - aplikací. Je třeba si uvědomit, že každou s těchto aplikací vytvářela jiná firma, za jiným účelem, pro jiné cílové prostředí atd. Integrátor se tedy vždy musí důkladně seznámit s koncepcí každé integrované aplikace, aby určil nejvýhodnější možnosti integrace, a poté detailně navrhnout a vyvinout rozhraní dané aplikace. Aplikace vždy, podle svého zaměření, vytváří nad daty tzv. náhledy, např. podle produktů, zákazníků, oddělení apod. Z hlediska procesní integrace je však třeba data

sledovat z pohledu protékajících podnikových procesů, což však málokterá aplikace umožňuje (a rovněž lidé, kteří s aplikací denně pracují a poskytují integrátorovi potřebné informace, nejsou schopni data v této úrovni vnímat). Sjednocení náhledů na data pro obě integrované aplikace, které je nezbytným předpokladem pro úspěšnou integraci, se nazývá sjednocení datových modelů aplikací. Integrátor je pak nucen při hledání integračního řešení pracovat za silných (tedy s velkým dopadem) předpokladů, které zvyšují riziko celého projektu.

Heterogenní prostředí je dáno zejména těmito rozdíly mezi aplikacemi [JURIC, 2002]:

a. Architektura aplikace (monolitická, dvouvrstvá, třívrstvá ([VOŘÍŠEK, 1997])
b. Objektově orientované, či procedurální řešení
c. Programovací jazyk
d. Úložiště dat (relační, hierarchické, objektové)
e. Různé technologie integračních rozhraní
f. Komunikační model (publish-subscribe, request-reply, konverzační)
g. Integrační řešení transakcí a zabezpečení
h. Způsob sdílení dat
i. Netypické formáty vyměňovaných dat

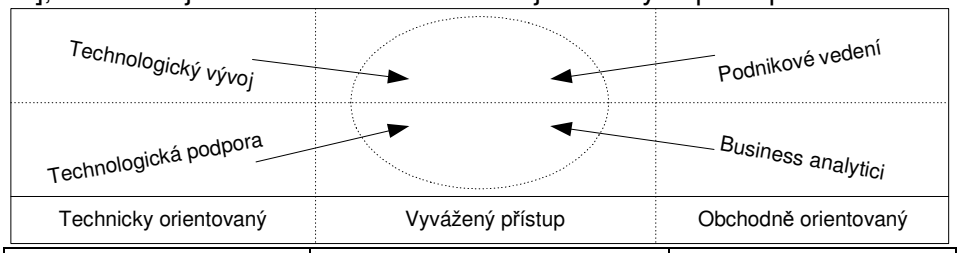
2. "Black-box" aplikace - prakticky v každém podniku existují aplikace, ke kterým chybí technická dokumentace (např. je napsána v nějakém exotickém jazyce), nebo neexistuje možnost vytvořit, či upravit rozhraní této aplikace (ať už z právních, finančních či jiných důvodů). Pro integrátora znamenají tyto aplikace složitý problém, ke kterému je třeba přistupovat individuálně. Z hlediska integrace rozlišujeme následující kategorie integrovaných aplikací dle jejich připravenosti pro integraci (od nejméně připravených):

   a. black-box aplikace - aplikace bez připravených (rozumějme i nedokumentovaných) rozhraní
   b. gray-box aplikace - black-box aplikace s určitou možností integrace (např. využitím webového výstupu)
   c. legacy aplikace - aplikace s předpřipravenými rozhraní s možností určitého přizpůsobení (tato rozhraní jsou často prodávána samostatně)
   d. white-box aplikace - otevřené aplikace s plně upravitelnými rozhraní

3. Možnosti a nároky aplikací na integraci - i pokud je aplikace je schopna poskytovat data požadovaná dalšími aplikacemi, je nutno řešit transformace těchto dat vzhledem k jejich struktuře a času (dávkové zpracování versus na vyžádání, doba odezvy apod.). Totéž samozřejmě platí i opačně i pro data vyžadovaná od ostatních aplikací. Tyto problémy se řeší na úrovni rozhraní i na úrovni integrační platformy. Způsob předání dat ze zdrojové aplikace do cílové aplikace označujeme pojmem integrační případ (viz kapitola 9 Integrační proces a integrační případ). Data zdrojové aplikace mohou vyžadovat rozdílné transformace do cílové aplikace, čímž často vzniká mezi dvěma aplikacemi více integračních případů ve stejném směru předávání dat.
4. Nedostatečně zmapované podnikové procesy - jak již bylo zmíněno, současná koncepce EAI vychází z analýzy podnikových procesů. Procesy však bývají popsány velmi vágně, neaktuálně, na hrubé úrovni, či jejich popis chybí zcela. Integrační projekty tak často začínají rozsáhlou analýzou (ve smyslu zmapování či BPR) podnikových procesů, čímž dochází k nárůstu nákladů na projekt ještě před zahájením vlastních integračních činností.

Řízení integračního projektu má samozřejmě značný vliv na jeho průběh, a proto pokud se snažíme charakterizovat tento typ projektů, musíme se seznámit i s častými chybami řízení těchto projektů. Touto problematikou se zabývá např. [LUBLINSKY, 2002], [MCGOVERN, 2003], [CARL, 2003]:

1. Chybný výběr integrační platformy v případě, kdy byl výběr integrátora včetně platformy proveden před zmapováním a vyhodnocením klíčových (rozumějme podporujících strategické podnikové cíle) podnikových požadavků. Výběr by měl být proveden až na základě závěrů úvodní studie.
2. EAI projekt je složitější, než se očekávalo. Tento stav nastává nejčastěji v případě, že zákazník podlehne slibům integrátora o rychlé a jednoduché integraci. Zákazník si musí uvědomit, že snahou integrátora je především projekt nastartovat do té fáze, kdy je pro podnik prakticky nemožné celý projekt zastavit.
3. Neúčast obchodních složek podniku na projektu. Tato oddělení často považují integraci za ryze technologickou záležitost, kterou pro ně vyřeší IT oddělení. U EAI projektů tomu však tak není vzhledem k tomu, že EAI vychází z podnikových procesů. Problémem se stává i vzájemná rivalita oddělení vzhledem k financování projektu, neboť integrační projekty se vždy pohybují na hranicích odpovědnosti těchto oddělení.
   [LAROIA, 2003] uvádí oba extrémní případy, kdy se EAI projektu věnují převážně lidé z IT nebo naopak jen lidé z businessu a doporučuje vyváženou účast obou složek - viz obrázek 5 - Možné přístupy k integračním projektům dle účasti (zdroj: [LAROIA, 2003], v tabulce je uvedena charakteristika jednotlivých přístupů.



Neflexibilní struktura	Kompromis mezi očekáváním a možnostmi	Chybný výběr technologie
"Technologie pro technologii"	Fázovaný přístup	Nerealistická implementační očekávání
Není vázáno na požadavky uživatelů	Soulad s podnikovými standardy	Úzkoprofilová architektura
Nejasné ROI	Měřitelné ROI	Nezohledněné stávající technologické investice

obrázek 5 - Možné přístupy k integračním projektům dle účasti (zdroj: [LAROIA, 2003])

4. **Neexistuje standardizovaná metodika implementace EAI**. Příčinou je to, že každý integrační projekt je odlišný v závislosti na integrovaných aplikacích, architektuře podnikového informačního systému a podnikových procesech - a obecné metodiky pro vývoj informačních systémů jsou zde nedostatečně konkrétní.
   Hlavní rozdíl spatřuji v tom, že ačkoli implementaci integrační platformy můžeme s dávkou abstrakce vnímat jako implementaci informačního systému, hlavní náplň projektu zůstává nepokrytá - vazba na okolí systému. Obecně používané metodiky jsou orientovány primárně interní vývoj aplikace a vazbu na okolní systémy vnímají pouze okrajově, nebo tento vztah vůbec neřeší.
   Dodavatel integrační platformy má zpravidla zpracovánu vlastní metodiku implementace, ale ta obsahuje převážně postupy, jak v dané platformě řešit typizované případy. Naopak neřeší celkový rámec projektu a výrazně zanedbává ostatní fáze projektu vyjma návrhu a vývoje. Teprve na technologické úrovni nalézáme výraznější míru standardizace, která se odvíjí od použitých technologií.
5. Přístupová oprávnění nejsou řešena v počátečních fázích. Každá aplikace má většinou nezávislý systém řešení přístupových práv, který je nutno rovněž integrovat s ostatními aplikacemi, na což se ovšem často zapomíná a řeší se teprve dodatečně.

6. EAI projekty se odlišují od klasických projektům implementace aplikací (což je třeba zohlednit při plánování a řízení projektu), zejména v těchto oblastech:
    a. Fáze analýzy a návrhu je výrazně náročnější, naopak fáze vývoje je relativně krátká.
    b. Úspěch tzv. POCu (proof-of-concept) má nižší vypovídací schopnost než u produktových implementací.
    c. Vyhodnocení úspěšnosti integrace aplikací (tj. celého projektu) je snazší, neboť je možné ji ověřit prakticky ihned po implementaci.
    d. Náročnost projektu roste exponenciálně, nikoli lineárně, díky počtu řešených vazeb mezi aplikacemi.
    e. Členové projektového týmu musí mít detailní znalosti mnoha odlišných typů systémů a technologií.
    f. Integrační projekty jsou charakteristické velkou mírou paralelizace - u větších projektů je třeba koordinovat několik samostatných týmů (často odlišně technologicky orientovaných), které pracují souběžně, a neustále hlídat vzájemné návaznosti.
    g. Vedle vývoje vlastní integrační platformy zasahují do projektu i integrované aplikace, resp. změny v nich. Prostředí, do kterého se integrační jádro implementuje, se neustále vyvíjí, proto je podstatnou součástí projektu i změnové řízení. V dnešní době si podnik nemůže dovolit zmírnit, natož zastavit, vývoj svých interních aplikací na dobu cca 1 roku, po kterou průměrně trvá nasazení nové integrační platformy.
7. Velký objem předávaných dat způsobuje problémy s výkonností. Návrhář rozhraní vždy zvažuje dvě hlediska - počet integračních bodů (např. funkcí poskytujících data) a objem poskytovaných dat integračním bodem. Existují dva možné extrémy, mezi kterými je třeba hledat kompromis:
    a. Jedna funkce poskytující vždy veškerá (tedy vyžadovaná alespoň jednou další aplikací) data aplikace vždy najednou - tato funkce je maximálně využitelná všemi aplikacemi (neboť každá si vybere jen svá data) a přináší úspory ve vývoji rozhraní. Na druhé straně při každém volání zatěžuje integrační platformu nadbytečnými daty a rozhraní se hůře udržuje (vzhledem k tomu, že je rozhraní sdílené, tak jsou na něj kladeny požadavky různorodých aplikací, což vede k různým duplicitám apod.).
    b. Mnoho funkcí, kde každá poskytuje pouze jeden údaj - aplikace tedy vždy dostane pouze data, o která si zažádá. Tento přístup je však náročný na vývoj a i na údržbu (je nutno podporovat velké množství funkcí a kontrolovat vzájemné závislosti).
8. Požadavky na monitorování a řízení procesů na integrační platformě jsou zpočátku projektu podceňovány. S koncepcí EAI je třeba si uvědomit, že je možno a nutno řídit i podnikové procesy, které propojují procesy nad integrovanými systémy. Řízení procesů by mělo být založeno na přesně definovaných metrikách, nazývaných souhrnně KPI (Key Performance Indicators).
9. Testování nového systému je velmi důležitou součástí projektu, u integračních projektů to platí dvojnásob. Integrační platforma je vždy kritickou komponentou podnikového informačního systému a navíc zde komunikují přímo systémy mezi sebou - pokud je něco špatně, pak to systémy narozdíl od lidí samy nepoznají (pouze zprostředkovaně) a ani nedokážou efektivně překlenout dobu do opravení chyby. Pro testování integračních projektů se zpravidla vyvíjejí různé simulátory integrovaných aplikací, případně i samotné testovací nástroje - a v nich se rovněž vyskytují chyby. Vedle toho chybí nástroje pro testování výkonnosti integrační platformy.

[KHANNA, 2004] se detailně zabývá **hlavními problémy integračních projektů**, které lze shrnout do těchto okruhů:

1. Chyby v zadání projektu - vyplývají z neznalosti EAI, chybných očekávání, špatně definovaného rozsahu projektu, nejasných požadavků. Tato metodika řeší tyto problémy v kapitolách 7 Business case a 8 Požadavky v integračních projektech.
2. Chyby v řízení projektu - příčinou je obecně chybějící standardizovaná integrační metodika, nedostatečná koordinace více týmů, chybějící spolupráce s aplikačními týmy, pozdní odsouhlasení testovacího plánu, nedostatek času na dokumentaci a školení, závislost na souběžné implementaci integrovaných aplikací, nedostatečné řízení změn. Tyto problémy řeší prezentovaná metodika v příslušných fázích projektu.
3. Chyby v řešení - způsobené výběrem nevhodných integračních nástrojů, technologií, postupů. Metodika se tímto zabývá v kapitolách 9 Integrační proces a integrační případ a 14 Výběr integrační platformy.
4. Nalezení a řešení chyb - nekompletnost testovacích příkladů (v metodice řešeno vazbou na integrační procesy), problematika simulace produkčního prostředí, migrace mezi prostředími, reprodukce produkčních chyb, chybějící definice kvality pro EAI (metodika na tyto problémy upozorňuje a vyžaduje je řešit dle konkrétních možností projektu).

2.2 Typy projektů

Existují různé typy integračních projektů, každý z nich má svou charakteristiku. V informační zdrojích jsem nenalezl definovanou typologii integračních projektů. V této kapitole tedy zavedeme výchozí členění EAI projektů s jejich charakteristikou.

Integračním projektem definuji projekt implementace informačního systému, jehož podstatnou součástí je řešení vybrané integrační úlohy. Integrační úloha řeší komunikaci mezi dvěma a více aplikacemi. Skupina úloh řešená integračními projekty je relativně omezená a pomůže nám při stanovení jejich základní typologie **dle řešené integrační úlohy**. Identifikoval jsem tyto integrační úlohy:

1. Integrace nově vytvořené aplikace do PIS - nová aplikace vyžaduje data z ostatních aplikací a rovněž zpracovaná data těmto aplikacím předává (řeší se tedy obousměrná integrace). Typickým příkladem z poslední doby byla vlna vytváření podnikových informačních portálů, u nichž se jednalo o principiálně nový typ aplikací, které vyžadovaly data z většiny podnikových aplikací a naopak nazpět předávaly klíčové informace pro více aplikací.
2. Nahrazení zastaralé aplikace novou aplikací - nahrazovaná aplikace je většinou pevně integrována s ostatními aplikacemi, což znamená provést důkladnou analýzu stávajících rozhraní, aby byly identifikovány potenciální dopady na okolní aplikace. Nová aplikace je většinou koncepčně řešena jinak, což znamená i odlišné řešení rozhraní na ostatní aplikace, tedy i výrazný zásah do těchto aplikací. Pokud je nahrazována některá z aplikací kritických pro podnikání (např. objednávkový systém), je třeba dále řešit i způsob "přepnutí" ze staré na novou aplikaci. V takovém případě lze volit mezi postupným přechodem (kdy po určité období běží obě aplikace současně), který je bezpečnější, avšak dražší, nebo tzv. boomem (kdy je stará aplikace odstavena a vzápětí nahrazena novou). U těchto projektů se náklady na vyřešení integrace mohou pohybovat až kolem 50 procent (expertní odhad) celého projektu nasazení nové aplikace.
3. Nahrazení, resp. upgrade integrační platformy - kompletní nahrazení integrační platformy za jinou představuje poměrně složitý projekt, ke kterému podnik přistupuje pouze v případě přechodu na koncepčně odlišný způsob integrace celého PIS, neboť v ostatních případech je riziko celého projektu neúměrně vysoké předpokládaným přínosům. Častějším projektem bývá modernizace (upgrade) integrační platformy, neboť nové verze často přinášejí technologicky odlišná řešení, která nejsou zpětně kompatibilní. V těchto případech se podnik musí rozhodnout, zda bude stávající rozhraní modernizovat tak, aby naplno využil nové technologie, nebo zda se spokojí s

tím, aby vše pracovalo jako dříve, pouze na novějších technologiích (v takovém případě se jedná pouze o "přeprogramování" stávajících rozhraní bez reálného dopadu na podnikání).

4. Koncepční změna integrace PIS - koncepční změnu integrace můžeme chápat například jako přechod z datové na procesně řízenou integraci, či nahrazení tzv. špagetové integrace (point-to-point koncepce) integrační platformou. V každém případě se jedná opět o poměrně složitý zásah do PIS. Jelikož se však v důsledku jedná o nahrazování jednotlivých rozhraní, či jejich skupin, lze tyto projekty často poměrně rozumně fázovat v menších etapách.
5. Modernizace rozhraní na aplikaci, resp. mezi aplikacemi - jedná se o běžný miniprojekt, kdy cílová aplikace požaduje od rozhraní dodatečnou funkcionalitu a je tedy potřeba funkcionalitu rozhraní doplnit. Tyto miniprojekty jsou většinou součástí většího projektu modernizace cílové aplikace.
6. Ostatní, atypické úlohy.

Prvotní implementace EAI probíhá nejčastěji jako samostatný projekt, jehož cílem je komunikačně propojit dvě, případně i více nejdůležitějších aplikací. Tento případ budeme uvažovat jako stěžejní pro vytvářenou metodiku. Sekundárním cílem takového pilotního projektu zároveň bývá sestavit detailní návod pro další aplikace, které budou integrovány v dalších, samostatných fázích. **Podle zkušeností podniku** s integrací aplikací tak můžeme projekt dále charakterizovat jako:

1. pilotní integrační projekt, nebo
2. rutinní integrační projekt.

Dalším důležitým kritériem je samotná **oblast integrace**, kde rozlišujeme (viz také [OVUM, 2001]):

1. Interní integrační projekty - interním integračním projektem označujeme všechny projekty, které řeší propojení několika aplikací v rámci jednoho podniku (např. předání informací z objednávkového systému do dalšího systému ke zpracování). Tyto projekty se označují zkratkou A2A (application-to-application).
2. Externí integrační projekty - externí integrační projekty řeší integraci aplikací mezi dvěma či více podniky (např. předávání telefonních čísel mezi telekomunikačními operátory v případě, kdy se zákazník rozhodne změnit operátora) a označujeme je zkratkou B2B (business-to-business).

Charakteristika, požadavky i průběh interních a externích integračních projektů se významně liší, a proto se v následujících kapitolách budeme zabývat **pouze interními integračními projekty**.

Podle charakteristických činností prováděných v integračním projektu rozdělujeme projekty do následujících skupin (viz také [OVUM, 2001]):

1. Agregační - integrace nové aplikace s mnoha různými stávajícími aplikacemi, kde dochází k agregaci mnoha informačních zdrojů do jednoho. Typickým příkladem je integrace podnikového informačního portálu.
2. Migrační - pročišťování IT infrastruktury, nebo migrace na vyšší verze produktů, kde dochází k normalizaci rozhraní, zavádění standardů, odstraňování nadbytečných rozhraní.
3. Procesní - automatizace podnikových procesů, kde hlavní důraz je kladen nikoli na aplikace, ale na vztahy mezi nimi. Ovlivňuje všechny integrované aplikace i uvnitř.

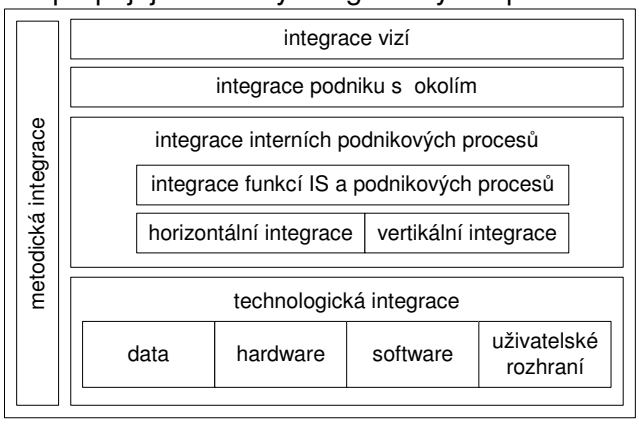
Mohli bychom používat i další členění integračních projektů, která by byla odvozena od členění obecných projektů implementace informačních systémů, jako je celková hodnota projektu, úspěšnost, délka trvání apod. Tato členění však nejsou směrodatná ve vztahu k zaměření této práce, proto se jimi dále zabývat nebudeme.

2.3 Obecné úrovně integrace

U projektů můžeme dále zkoumat **úroveň integrace**, která je aplikována. Při rozsáhlých projektech se zpravidla realizuje integrace na všech dále uvedených úrovních. Komplexní souhrn úrovní integrace při implementaci informačního systému uvádí ([VOŘÍŠEK, 1997], str. 107), kde z 5 základních úrovní integrace (obrázek 6 - Úrovně integrace informačního systému (zdroj: [VOŘÍŠEK, 1997])) jsou pro námi zkoumanou oblast vnitropodnikové integrace podstatné následující úrovně:

1. Integrace podnikových procesů, která zajišťuje provázanost podpory podnikových procesů aplikacemi i vně samotných aplikací (tedy pomocí rozhraní). V důsledku integrace podnikových procesů se integrační platforma zpravidla stává manažerem celého podnikového procesu (k čemuž dochází v případě, že celý proces je podporován více aplikacemi a žádná z nich není v procesu dominantní), což na ni klade dodatečné nároky. Na úrovni procesní integrace je nutno vždy zvažovat, které podnikové procesy je výhodné integrací podporovat, a které nikoli, neboť v praxi se můžeme setkat s procesy, které lze s nižšími náklady podporovat manuálně (typicky procesy bez požadavků na dobu zpracování (odezvy), procesy s malou přidanou hodnotou a zároveň se složitými business pravidly, řídce spouštěné procesy s malým objemem informací, atd.).
2. Technologická integrace zahrnující
   - integraci datovou, která zajišťuje zejména transformaci struktur dat mezi aplikacemi a předávání dat mezi aplikacemi,
   - softwarovou, zahrnující vzájemné softwarové propojení aplikací,
   - hardwarovou, která pokrývá sjednocení hardwarových komponent do jednotné počítačové sítě podniku,
   - uživatelského prostředí, která sjednocuje ovládací principy jednotlivých aplikací.

   Z těchto vyjmenovaných úrovní je pro EAI významná především úroveň datové integrace a částečně i uživatelského rozhraní ve smyslu sjednocení prezentační vrstvy.
3. Metodická integrace spojuje všechny metody, techniky a nástroje, které se používají v ostatních úrovních systémové integrace. V prostředí integrace se jedná zejména o prvky, které zavádějí standardy (při analýze, návrhu, vývoji, provozu atd.) do vztahů mezi aplikacemi a propojují metodiky integrovaných aplikací s integrační platformou.



obrázek 6 - Úrovně integrace informačního systému (zdroj: [VOŘÍŠEK, 1997])

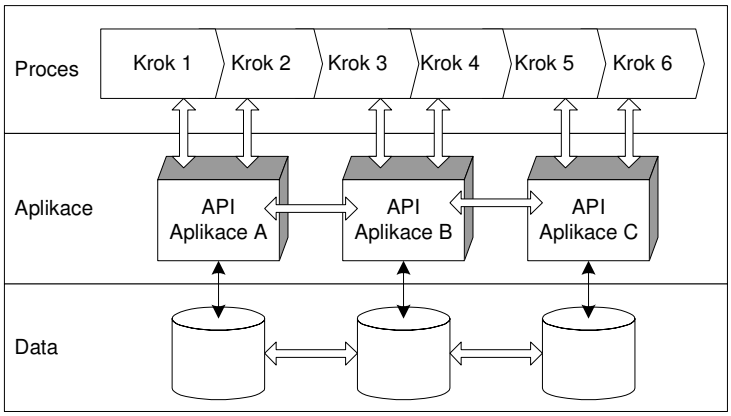
V kapitole 10 Integrační postupy si budeme definovat odlišné členění, které se užívá v integračních projektech. Zde uvádím orientační terminologické mapování mezi těmito úrovněmi:

1. Integrace podnikových procesů => integrace procesní.
2. Integrace technologická datová => integrace datová.

3. Integrace technologická softwarová, prostřednictvím rozhraní aplikací => integrace služeb (též aplikační, funkční, API, objektová).
4. Integrace technologická, uživatelského prostředí => integrace prezentační (též UI, GUI, front-endová).

Následující obrázek 7 - Vztah mezi úrovněmi integrace (zdroj: [SIFTER, 2003]) zachycuje základní kombinaci integračních úrovní, kde je provedena integrace na samostatných úrovních - datové a aplikační. Procesní integrace nemůže existovat bez podpory na úrovni aplikací a pouze přebírá odpovědnost za řízení aplikační integrace. Aplikační integrace může existovat bez vazby na datové zdroje (v případě žádosti o provedení operace, nikoli žádosti o data), ale v praxi se takto nezávisle nevyskytuje.

Obrázek nezachycuje integraci uživatelského rozhraní, kterou bychom mohli považovat za variantu aplikační integrace, pokud budeme považovat za GUI za specifické aplikační rozhraní. Reálně se jedná o relativně nezávislou a atypickou úroveň integrace, která není využívána při procesní integraci. V praxi se můžeme dále setkat s přímou vazbou mezi procesní a datovou úrovní, kdy je k dispozici pouze databáze s údaji, avšak neexistuje žádná aplikace, která by byla schopná poskytnout své rozhraní pro provedení integrace nad těmito údaji. V takových případech poskytuje integrační platforma jakousi pseudoaplikaci, což je ovladač na příslušnou databází, který je schopen data poskytovat.



obrázek 7 - Vztah mezi úrovněmi integrace (zdroj: [SIFTER, 2003])

3. Výběr mateřské metodiky

V této kapitole je nejprve popsána logická souvislost mezi hledáním specifických rysů integračních projektů a tvorbou metodiky, dále pak pátrání po vhodné metodice pro integrační projekty - prostřednictvím kategorizace množiny metodik používaných pro integrační projekty, anketou u vybraných firem a vlastním zkoumáním projektových metodik. Výsledky pátrání vedly k rozhodnutí zvolit mateřskou metodiku, z které bude odvozena vlastní metodika pro integrační projekty. Výběr MMDIS jako mateřské metodiky je zdůvodněn v závěru kapitoly.

Odpověď na otázku, jaký postup zvolit při snaze charakterizovat projekty zabývající se integrací aplikací, nebyla jednoduchá. Nosná logická úvaha vedla k tomu, že strukturované zkoumání projektu znamená detailně analyzovat jeho jednotlivé fáze. Je základní vlastností projektové metodiky tyto fáze specifikovat (ať už ve formě lehké či těžké metodiky). Vzhledem k tomu, že primárním smyslem projektové metodiky je definovat doporučené řešení specifických rysů dané skupiny projektů, lze z ní zpětně dovodit tyto specifické rysy. Tento způsob charakterizace projektů jsem vyhodnotil jako kvalitativně nejvýhodnější a přitom realizovatelný, neboť metodiky jsou průběžně ověřovány a zdokonalovány na řadě reálných projektů a poskytují tak reálné podklady.

Metodologii (metodiku) definuje [CHLAPEK, 1997] jako "doporučený souhrn etap, přístupů, zásad, postupů, pravidel, dokumentů, řízení, metod, technik a nástrojů pro tvůrce informačních systémů, který pokrývá celý životní cyklus informačních systémů. Definuje kdo, kdy, co a proč má dělat."

Při identifikaci charakteristických rysů integračních projektů jsem postupoval v několika krocích. Nejdříve bylo potřeba zjistit, jakým způsobem probíhá sledovaná projektová fáze v integračním projektu - to bylo možné kombinací pozorování reálných projektů, vyhledáváním informací v informačních zdrojích či analýzou existující integrační metodiky. Porovnáním se stejnou projektovou fází obecného projektu jsem identifikoval odlišné činnosti či artefakty, které jsou v práci zaznamenány a tvoří základ prezentované specifické metodiky metodiky. Posledním krokem bylo dohledání příčin, které vedly k existenci nalezených odlišností, a ty jsou hledanými charakteristikami integračních projektů.

Tím se problém přenesl na získání vypovídající metodiky pro integrační projekty a vhodné obecné metodiky pro provádění komparace. Tento přístup vede rovněž k tomu, že bude-li srovnání děláno důsledně vůči jedné obecné metodice, pak ve specifické metodice (dále ji budu nazývat dceřinou metodikou) stačí definovat pouze rozdíly vůči této obecné metodice (dále ji budu nazývat mateřskou metodikou), neboť u zbývajících částí se lze odkazovat na mateřskou metodiku.

Jelikož obecná projektová metodika mi mohla poskytnout pouze celkový rámec projektu a jeho základní strukturu, očekával jsem, že pozorováním a ověřováním integračních projektů v této struktuře a rámci dokáži identifikovat jejich typické rysy, které budu moci navázat na strukturu mateřské metodiky, a tím vytvořit metodiku odvozenou (dále ji budu nazývat dceřinou), která bude specifická přímo pro integrační projekty.

Nejdříve jsem zkoušel najít metodiku, která by již byla navržená přímo pro integrační projekty. Při hledání jsem identifikoval tyto skupiny metodik:

1. Obecné metodiky vývoje informačních systémů - některé jsou veřejně publikované, avšak nejsou specifické pro oblast EAI.
2. Metodiky implementace konkrétní integrační platformy - zpravidla poloveřejné (dostupné za určitých podmínek, např. pro realizaci konkrétního projektu), výrazně závislé na konkrétním produktu, jedná se spíše o souhrn specifických metod, než o komplexní metodiku; do této kategorie zahrnuji také metodiky integrace postavené na konkrétní technologii (např. .NET).

3. Metodiky integrace vybraného produktu - zpravidla poloveřejné (dostupné za určitých podmínek, např. pro realizaci konkrétního projektu), zcela závislé na konkrétním produktu, popisující integraci pouze jednoho produktu (např. Siebel).
4. Metodiky implementace EAI - veřejně nedostupné know-how firem zabývajících se systémovou integrací. Tato skupina se nejvíce blíží ke stanovenému cíli, avšak přístup k těmto informacím je velmi obtížný. U těchto metodik lze předpokládat určitou neaktuálnost ve vztahu k novým technologiím a koncepcím, silnou závislost na integračních platformách přeprodávaných danou firmou a skryté know-how v konkrétních zaměstnancích.

Pro skupinu č. 4 jsem realizoval i malou anketu u firem zabývajících se integrací. Dle žebříčku TOP 10 systémových integrátorů 2005 [TOP10] jsem oslovil 10 firem s mailem přiloženým na obrázku 8 - Průzkum integračních technologií u TOP 10 SI (zdroj: autor).

Komu: info@autocont.cz, deltax@deltax.cz, info.praha@eon-is.cz, info.cz@hp.com, info@i.cz, ids@ids-scheer.cz, marcom.cee@logicacmg.com, ness@ness-europe.com, Vera.Hnatova@pvt.cz, info.cz@siemens.com
Od: Jan Karas/KIT/VSE
Datum: 20.12.2005 12:33
Předmět: Pruzkum integracnich metodologii u TOP10 SI

Dobrý den,

jsem studentem postgraduálního studia na VŠE v Praze v oboru Informatika a ve své dizertační práci se zabývám řízením
projektů vnitropodnikové integrace aplikací. Dovolil jsem si oslovit firmy z aktuálního žebříčku TOP 10 systémových
integrátorů a požádat Vás o následující informace:

1) Realizujete projekty zaměřené na vnitropodnikovou integraci aplikací?
Pokud ano,

2) používáte pro řízení těchto projektů vlastní firemní metodologii nebo obecně dostupnou metodologii?

3) používáte pro řízení těchto projektů metodologii specificky upravenou pro integraci aplikací nebo obecnou metodologii pro vývoj informačních systémů?

4) Můžete uvést název metodologie, kterou používáte?

Výsledky tohoto průzkumu zpracuji anonymně a budou užity výhradně pro potřeby mé dizertační práce. Děkuji předem za Váš čas a poskytnuté informace. Informujte mne, prosím, i v případě, že si nepřejete odpovídat. O zaslání odpovědí prosím do 5.1.2006.

S pozdravem
Jan Karas

obrázek 8 - Průzkum integračních technologií u TOP 10 SI (zdroj: autor)

Do uzavírky průzkumu mi odpověděly čtyři oslovené firmy. Tyto firmy mi shodně odpověděly, že realizují projekty vnitropodnikové integrace, **používají vlastní firemní metodiku** (ve třech případech kombinovanou s obecnou metodikou) a **tyto metodiky jsou obecné pro vývoj informačních systémů** (v jednom případě je často kombinována s metodikou dodavatele integrační platformy) a v jednom případě se používá firemní metodika pro integrační projekty. Mezi používanými metodikami byly zmíněny metodiky MMDIS, LBMS, PRINCE, Chestra, Cortex, ASAP (metodika pro implementaci SAPu). Dále se mi podařilo dohledat na firemním webu údaje o páté firmě, která používá metodiky PRINCE a ASAP.

Stejné závěry jsem získával i v období, které předcházelo tomuto průzkumu, z rozhovorů s kolegy z dalších firem. Stejně tak v obou firmách, v kterých jsem se pracoval na integračních projektech, se používaly obecné firemní metodiky pro vývoj informačních systémů, CATALYST a LBMS.

Domnívám se, že příčin malého počtu odpovědí bylo několik:

- Dotazy se nedostaly ke kompetentním osobám (neznalost problematiky, chybějící pravomoc poskytnout požadované informace). Ze všech firem jsem ovšem dostal doručenku potvrzující přečtení mailu.
- Odpovědi na položené otázky jsou považovány za obchodní tajemství nebo odpovědi firmy považovaly za kompromitující.
- Účast v průzkumu nebyla pro oslovené firmy zajímavá ve vztahu k jejich podnikání.

Vlastním průzkumem informačních zdrojů jsem ověřoval dostupné metodiky, zda by se daly využít jako specifické pro integrační projekty. Zabýval jsem se těmito metodikami:

1. Obecné metodiky vývoje IS
    a. globální - RUP [KRUCHTEN, 2003], PRINCE2, MMDIS [VOŘÍŠEK, 1997]
    b. vývojové tradiční - SSADM, Code-and-Fix, V-Model, Waterfall, Spiral, Reverse Engineering (přehledově popsány v [CHARVAT, 2003])
    c. vývojové agilní ([COCKBURN, 2001]) - XP ([WAKE, 2001], [JEFFRIES, 2000], [BECK, 2000]), RAD
    d. firemní - Catalyst, LBMS
2. Metodiky implementace konkrétní integrační platformy - BEA Systems, TIBCO, Oracle
3. Metodiky integrace vybraného produktu - Siebel (UAN), SAP (ASAP)
4. Metodiky implementace EAI - interní firemní know-how firem CSC Computer Sciences a NESS CEE (např. [CSC, 2001], [LANGER, 2004])

Mnoho dalších je uvedeno např. v [CHARVAT, 2003] a [BUCHALCEOVÁ, 2005], ale žádná z nich nebyla adresována na integrační projekty.

Během zkoumání se mi však nepodařilo nalézt metodiku zaměřenou na integrační projekty, která by nevykazovala některý z následujících **nedostatků**:

1. Metodika nezohledňuje specifické potřeby integračních projektů.
2. Metodiku nelze použít univerzálně na integrační projekt (např. závislost na určitém produktu).
3. Metodika není dostatečně zdokumentována (know-how v lidských zdrojích).
4. Metodika je dostupná pouze velmi omezené skupině lidí (např. komerční metodika).
5. Metodika nepokrývá celý životní cyklus projektu (např. pouze vývoj).

[BUCHALCEOVÁ, 2005] uvádí podobné obecné problémy použití metodik v projektech:

- Metodiky nejsou dostatečně a jednotně popsány ani rozumně kategorizovány.
- Nejsou definována kritéria pro výběr vhodné metodiky pro určitý typ projektu ani postupy pro její přizpůsobení na konkrétní podmínky firmy a projektu.
- Procento firem v ČR využívající metodiku je nižší než ve světě. Hlavní příčiny spatřuje v chybějící lokalizaci metodik do českého jazyka a malou ochotou českých firem investovat do komerčních metodik.

S tímto výsledkem jsem se rozhodl vybrat mateřskou metodiku a z ní vytvořit dceřinou metodiku, která bude zohledňovat specifické rysy integračních projektů. Tím, že jsem se rozhodl pro zdědění všech charakteristik z mateřské metodiky, jsem se nadále nemusel zabývat budováním celého rámce metodiky jako takové, ale soustředil jsem se pouze na odlišné charakteristiky integračních projektů. Mateřská metodika musí mít takové vlastnosti, aby jejich zděděním do dceřiné metodiky nedošlo ke kolizi s potřebami integračního projektu.

Mateřskou metodiku jsem se rozhodl zvolit z první skupiny (obecné metodiky vývoje IS) zkoumaných metodik, neboť u 2. a 3. skupiny (metodiky implementace konkrétní integrační platformy a metodiky integrace vybraného produktu) se mi jeví jako velmi obtížné

až nemožné se zbavit závislosti na konkrétním produktu a u 4. skupiny (metodiky implementace EAI) by nebylo možné publikovat výslednou metodiku bez podstatných omezení.

Za mateřskou metodiku jsem si **zvolil MMDIS** (Multidimensional Management and Development of Information System), neboť

- pokrývá celý životní cyklus projektu
- má výrazně strukturalizovaný přístup k jednotlivým etapám projektu, což umožňuje s touto metodikou dále snadno pracovat a rozvíjet ji
- je referenční metodikou pro KIT (Katedry informačních technologií VŠE Praha)
- principiálně nekoliduje z potřebami integračních projektů, naopak je podporuje svými základními principy

Princip metodiky je dle [VOŘÍŠEK] "myšlenkový přístup k chápání a analýze problému a s ním spojené zásady (pravidla) řešení problému". MMDIS deklaruje (viz [VOŘÍŠEK, 1997], [VOŘÍŠEK], [BUCHALCEOVÁ, 2005]) těchto svých 11 principů - multidimenzionalita, vrstevnatost, integrace, flexibilita, otevřenost, standardizace, kooperace, procesní pojetí, učení a růst (postupné zlepšování procesu), lokalizace zdrojů a rozhodnutí, měřitelnost. MeFIS, který vychází z MMDIS, akceptuje 8 z těchto principů a přidává 5 vlastních - sladění IS/ICT s podnikovými procesy, orientace na služby s podporou globální architektury (SOA), rozlišování úrovní abstrakce, prvořadá úloha lidí, modelem řízené budování IS/ICT.

Integrační metodika není s žádným z těchto principů ve sporu, naopak většinu těchto principů sama používá. Některé z nich však přímo neuplatňuje (pouze zprostředkovaně přes mateřskou metodiku) - lokalizaci zdrojů, měřitelnost a prvořadou úlohu lidí. Princip orientace na služby chápe jako jeden z prvořadých integračních konceptů, který však není aplikovatelný za všech podmínek. Za klíčové principy pro integraci aplikací považuje otevřenost metodiky pro nové integrační postupy, práci v heterogenním aplikačním prostředí a úzkou provázanost na podnikové procesy skrze integrační procesy.

Při popisu dceřiné metodiky jsem se nechal inspirovat způsobem popisu uvedeným v [NOVOTNÝ, 2005], který používá podobný přístup pro metodiku a charakteristiku projektů implementace business inteligence.

4. Životní cyklus integračního projektu

V této kapitole se podrobně seznámíme s životním cyklem integračního projektu. Fáze projektu vycházejí z mateřské metodiky MMDIS a definují rámec pro následující kapitoly popisující jednotlivé části fází detailně.

Životní cyklus se zpravidla dělí do několika, časově i obsahově na sebe navazujících fází. Metodika umožňuje rychlou orientaci v projektu a jeho efektivní řízení. Je třeba mít stále na paměti, že žádná metodika negarantuje úspěch projektu, ale nabízí principy a postupy, které mohou vést k nejlepším řešením. Používané metodiky se zpravidla mírně odlišují ve stanovených hranicích jednotlivých fází, nikoli však jejich rámcovým obsahem, či návazností mezi fázemi.

V informačních zdrojích jsem nalezl dva články, které se zabývají fázemi integračních projektů. [LAROIA, 2003] definuje základní projektové fáze EAI projektu, a to následovně:

1. Koncepční fáze - definuje strategické cíle, přístup (koncepční myšlenky), rámec projektu.
2. Fáze požadavků - analyzuje a vytváří logickou (technologicky nezávislou) infrastrukturu EAI integračních komponent včetně popisu rozhraní.
3. Fáze specifikace - analýzou požadavků identifikuje sdílené business objekty (komponenty) s cílem redukovat složitost projektu. Probíhá detailní specifikace rozhraní v této struktuře:
    a. business účel,
    b. technický koncept,
    c. mapa rozhraní,
    d. zadání pro vývoj,
    e. testování.
4. Fáze vývoje - vytváří se jednotlivé komponenty a rozhraní a probíhá intenzivní testování.

Ve srovnání s námi zvolenou metodikou MMDIS tento přístup sloučil fáze GST, IST a US do koncepční fáze, GAN se shoduje s fází požadavků, DAN s fází specifikace, IM s fází vývoje, ZA a PU nejsou pokryty.

[SCHMIDT, 2003d] zmiňuje možnost využít pro fázi DAN postupy z extrémního programování (XP), zejména vzhledem k častým změnám požadavků na rozhraní. Tento přístup je charakterizován takto:

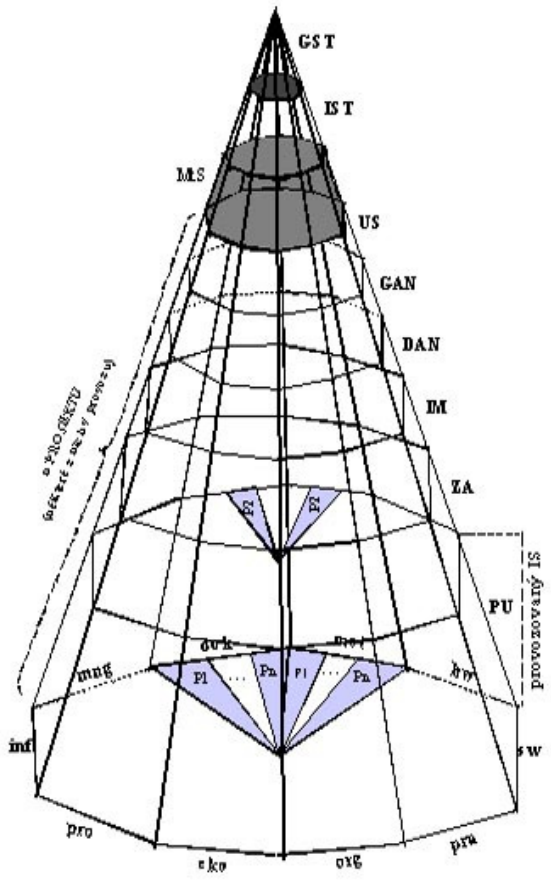
1. Uživatelé definují požadavky na základě příkladů a scénářů.
2. Tým používá společný systém názvů a popisů.
3. Vývojáři připravují nejdříve moduly pro automatické testování, teprve pak píší samotný kód.
4. Vývojáři často přepracovávají napsaný kód.
5. Všichni vývojáři pracují na celém společném kódu.
6. Programuje se podle společného programovacího standardu, takže výsledný kód neobsahuje individuální odchylky.

V praxi se mi nepodařilo ověřit aplikaci technik extrémního programování v integračních projektech, proto tento přístup nemohu hodnotit. [SCHMIDT, 2003b] považuje tyto techniky za vhodné pro vývoj rozhraní aplikací vzhledem k potřebě jejich neustálého upravování. Prezentovaná metodika pro integrační projekty aplikaci těchto technik nijak neomezuje.

V této kapitole vyjdeme ze zvolené mateřské metodiky MMDIS [VOŘÍŠEK, 1997], která definuje následující etapy projektu (viz obrázek 9 - Konceptuální model MMDIS (zdroj: [VOŘÍŠEK, 1997])):

1. Globální strategie (GST) - společně s informační strategií zajišťují první tři úrovně systémové integrace (integrace vizí, integrace podniku s okolím, integrace interních podnikových procesů).
2. Informační strategie (IST) - rozšiřuje výstupy globální strategie o úroveň technologické integrace, celkovou architekturu informačního systému, definuje jednotlivé projekty a zajišťuje jejich konzistenci.
3. Úvodní studie (US) - detailně posuzuje realizovatelnost projektu (resp. požadavků na něj) a zpracovává variantní návrhy řešení.
4. Globální analýza a návrh (GAN) - vymezuje hlavní funkce a data na konceptuální úrovni, je nezávislá na implementačním prostředí.
5. Detailní analýza a návrh (DAN) - transformuje konceptuální návrh do úrovně technologické.
6. Implementace (IM) - představuje fyzickou realizaci návrhu (vývoj), testování a kompletaci dokumentace.
7. Zavádění (ZA) - systém se instaluje, migrují se data z nahrazovaných systémů, školí se uživatelé.
8. Provoz a údržba (PU) - systém se využívá v reálném prostředí a přizpůsobuje se novým požadavkům.

Poznámka: [VOŘÍŠEK] přidává ještě poslední fázi Vyřazení (VY), v které je činnost systému ukončena. Tato fáze je však zmíněna jen okrajově, proto se jí nebudeme dále věnovat.



obrázek 9 - Konceptuální model MMDIS (zdroj: [VOŘÍŠEK, 1997])

Typ integračního projektu nemá z hlediska popisu životního cyklu zásadní význam, jelikož každý úspěšný projekt musí projít všemi vyjmenovanými etapami. Odlišnosti se objevují v samotném obsahu jednotlivých fází, který se rovněž pokusíme rámcově vymezit, a

proto si jako **vzorový projekt zvolíme procesní typ projektu s cílem nahrazení integrační platformy**. Důvodem pro tuto volbu bylo, že jej považuji za nejkomplexnější integrační projekt, který nám umožní upozornit na klíčové prvky každé fáze. Tento typ projektu na některých místech rozšířím na typ projektu, v kterém se implementuje nová aplikace a která vyžaduje implementaci integrační platformy (tzv. nadřízený projekt).

4.1 Globální strategie

Není snadné nalézt přímou vazbu mezi globální strategií podniku a integračním projektem. Důvodem je, že integrační platforma sama o sobě není zpravidla přímým zdrojem podnikových zisků. Přesto lze takové pojítko najít zprostředkovaně přes informační strategii a zejména v těchto případech:

- Globální strategie obsahuje rozhodnutí implementovat podnikový proces, který bude vyžadovat přímou podporu nové aplikace. Pro implementaci této nové aplikace bude nutné ji integrovat s ostatními podnikovými aplikacemi prostřednictvím integrační platformy.
- Globální strategie obsahuje rozhodnutí o implementaci nového podnikového procesu, který bude řízen přímo integrační platformou (konkrétní případ je zmíněn i v kapitole 17.6.3 Projekt 3 - Aplikace pro zřizování služby (2004, 2006)).
- Globální strategie požaduje optimalizaci podnikových procesů a integrační platforma bude poskytovat informace o efektivnosti procesů (prostřednictvím KPI).

4.2 Informační strategie

Integrační projekty se zřídka vyskytují jako zcela nezávislé samostatné projekty (pouze některé příklady migračních typů projektů), zpravidla se s nimi setkáme spíše v roli subprojektů větších projektů zaměřených na implementaci vybrané aplikace, či podnikového procesu. Z toho plyne, že integrační projekty přebírají cíle globální a informační strategie z nadřízeného projektu. Cíle integračního projektu můžeme tedy obecně definovat následovně (dle předpokládaných priorit):

1. Maximální podpora nadřízeného projektu (tedy požadavků implementované aplikace či procesu)
2. Dodržení interních standardů pro integraci (koncept integrace, integrační platforma včetně technologií, atd.)
3. Minimalizace nákladů na integraci nadřízeného projektu

Součástí informační strategie by měla být **integrační strategie** a veškeré integrační projekty by ji měly respektovat. O integrační strategii pojednává kapitola 10.3 Integrační koncepce. Integrační strategie by měla definovat hlavní integrační cesty, o který pojednává kapitola 11 Model komplexní integrované architektury PIS.

Informační strategie definuje jednotlivé projekty, mezi nimi i integrační projekty. Ke každému projektu se zpracovává dokument nazývaný business case (obchodní plán), který je podkladem pro rozhodnutí sponzora projektů o realizaci či nerealizaci projektu. O přípravě business case dokumentu k integračnímu projektu pojednává samostatná kapitola 7 Business case. Z definice projektu se odvozují požadavky a akceptační kritéria projektu, o kterých píšu v kapitole 8 Požadavky v integračních projektech.

4.3 Úvodní studie

Je-li projekt schválen k realizaci, nastává další fáze - příprava úvodní studie, často označované jako studie proveditelnosti (feasibility study). Ta v integračním projektu vychází zejména z:

1. Cílů definovaných ve fázi globální a informační strategie, přesněji řečeno z projektových požadavků na integrační platformu.

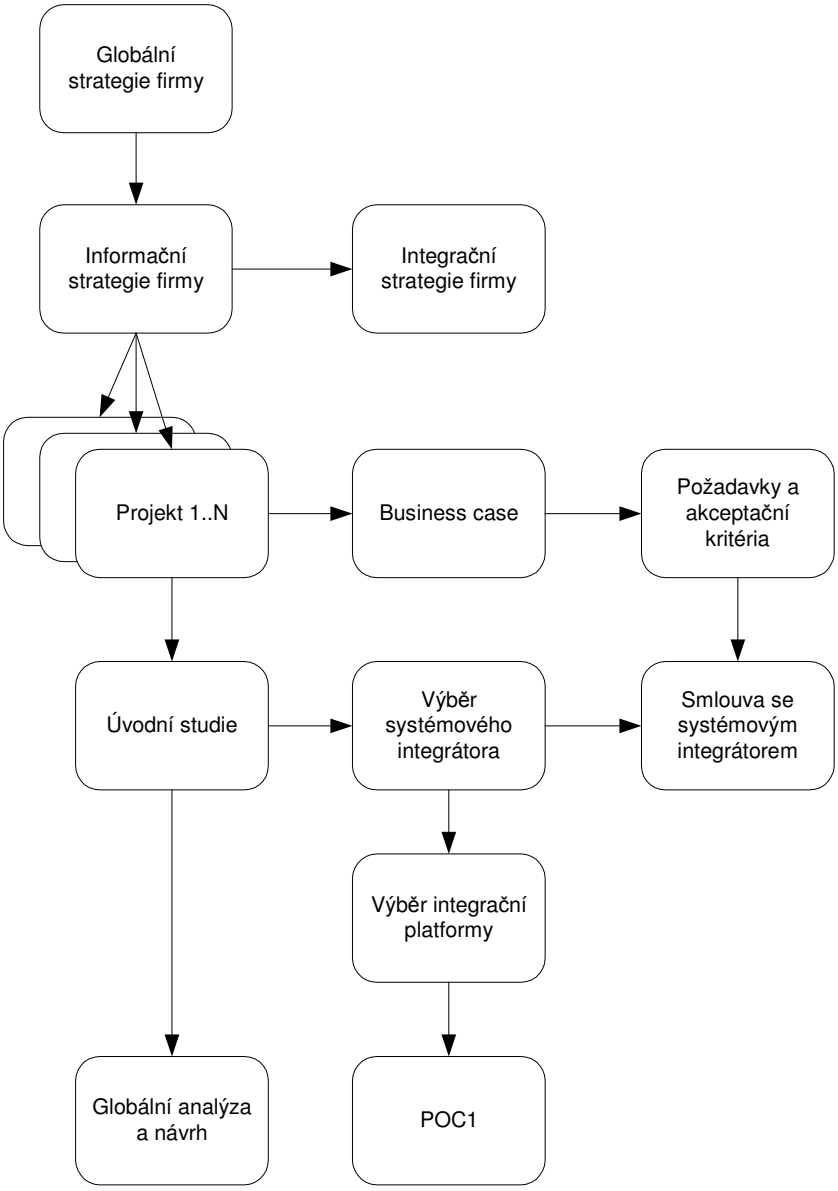
2. Požadavků nadřízeného projektu (viz cíl č. 1), které by měly obsahovat informace o:
    a. Datech, s kterými bude implementovaná aplikace pracovat (s označením tzv. kritických dat, tedy skutečně nutných).
    b. Datech, která bude aplikace primárně udržovat (v tom smyslu, že bude jejich zdrojem pro ostatní aplikace) včetně kritických nároků ostatních aplikací na tato data (objemy dat, perioda aktualizace, atd.).
    c. Datech, které bude vyžadovat od ostatních aplikací (a které aplikace jsou primárním zdrojem těchto dat) včetně kritických nároků na rozhraní (objemy dat, perioda aktualizace, atd.).
    d. Stávajících podnikových procesech, které se projektu týkají, a předpokládaných změnách.
    e. Podnikových procesech, které budou aplikací podporovány a jakým způsobem.
    f. Aplikacích, které budou nahrazeny novou aplikací, případně budou výrazně upraveny.

    Poznámka: Data se na této úrovni popisují pouze jednoznačnými názvy entit.
3. Možností integrační platformy.
4. Možností implementované aplikace (především informace o připravených a nebo podporovaných způsobech integrace.
5. Soupisu předpokládaných rozhraní s přiřazenou prioritou jejich důležitosti z hlediska projektu.
6. Časové posloupnosti projektu (projektového plánu), kdy budou rozhraní požadována. Z hlediska integračního projektu je principiálně přirozené řešit jednotlivá rozhraní (integrační případy) postupně dle připojovaných aplikací, avšak pro nadřízený projekt je naopak přirozené implementovat postupně podporu pro jednotlivé podnikové procesy (což často znamená řešit integrační případy mezi více aplikacemi). Ve výsledku je proto nutné najít kompromis mezi oběma přístupy.
7. Typů jednotlivých aplikací (viz kapitola 2.1 Obecná charakteristika integračních projektů) z hlediska integrace.

Finálním výstupem úvodní studie by měl být dokument (úvodní studie)

1. mapující předpokládaná rozhraní, pokud možno rozčleněná na úrovni integračních případů,
2. doplňující projektový plán nadřízeného projektu s popsanými závislostmi (obsahující rovněž rozhodnutí o způsobu přechodu na nový systém, tj. postupnou nebo jednorázovou migraci s nebo bez souběhu stávajícího systému). Projektový plán již zohledňuje alokaci potřebných zdrojů i datumy dodávek od systémového integrátora.
3. Popisující předpokládané využití integrační platformy (základní představu o architektuře řešení dle modelu integračních cest v kapitole 11 Model komplexní integrované architektury PIS),
4. popisující předpokládané dopady na integrované aplikace,
5. identifikující kritické faktory (ve smyslu zvyšujících riziko projektu),
6. shrnující předpokládanou náročnost (zejména finanční) integračního projektu za vyjmenovaných předpokladů (často se zpracovává několik variant). Podkladem pro tyto odhady jsou nabídky firem, v roli systémových integrátorů, na realizaci projektu. Součástí výběrového řízení je i výběr integrační platformy pro potřeby projektu. Proces jejího výběru je popsán v samostatné kapitole 14 Výběr integrační platformy. U velkých projektů je součástí nabídky i prezentace dané platformy, kde mohou firmy prezentovat i tzv. POC1 (proof-of-concept, demo) s vybranou základní funkcionalitou.

Dle výsledků úvodní studie se znovu rozhoduje o zrušení projektu. Pokud projekt i nadále pokračuje, pak se uzavírá smlouva se systémovým integrátorem, čímž se současně upřesňuje a fixuje zadání projektu.



obrázek 10 - Fáze GST, IST, US v integračním projektu (zdroj: autor)

4.4 Globální analýza a návrh

Etapa globální analýzy a návrhu je období koncepčních rozhodnutí. Globální analýza přebírá závěry úvodní studie a dále je rozpracovává. Zejména požadavky na integrační procesy a možnosti jejich implementace na konceptuální úrovni.

Jako příklad si uveďme požadavek na předávání změn v údajích o zákazníkovi do 60 sekund z CRM (Customer Relationship Management) do platebního systému. U takového procesu tedy zjistíme, jaké údaje CRM poskytuje a jaké platební systém vyžaduje. Pokud některá požadovaná data chybí, pak musíme zjistit jejich zdroj a rovněž jej analyzovat. Dále pak, jakým způsobem probíhá změna těchto údajů v CRM a přes jaké rozhraní CRM lze tyto údaje předat a naopak zapsat do platebního systému. Poté navrhneme, kterou integrační cestou budeme data předávat - zde bych například navrhnul cestu z CRM přes integrační server do platebního systému a jako záložní variantu z CRM přes ETL do ODS a z ODS přes ETL do platebního systému.

Při analýze integračních procesů zkoumáme současné i nové vazby mezi dotčenými aplikacemi (nejen automatizovanými, ale i manuálními) za účelem konečného zmapování všech datových toků mezi aplikacemi relevantními pro daný projekt. Identifikované toky posléze seskupujeme do skupin dle integrovaných aplikací, čímž si definujeme požadavky na rozhraní aplikací.

Z výše uvedeného příkladu bychom identifikovaly požadavky na dva adaptéry CRM a dva pro platební systém (vždy jeden pro komunikaci s integračním serverem a druhý pro ETL, resp. ODS). Zároveň bychom do těchto adaptérů přidaly i další smluvní požadavky na synchronizaci produktových instancí zákazníka z dalších integračních procesů.

V rámci globálního návrhu jsou pro tyto skupiny stanoveny integrační cesty, které jsou směrodatné v následující fázi detailní analýzy a návrhu (u každé skupiny je např. částečně odlišná struktura a obsah detailní analýzy, neboť je třeba klást důraz na jiné prvky integrace).

Specifickou skupinou jsou integrační případy řešené předpřipravenými konektory aplikací. Tímto přístupem se snadno odhalují společné komponenty celého řešení, což v důsledku zvyšuje efektivitu projektu a umožňuje i snazší následnou údržbu. Výstup globálního návrhu by tedy měl obsahovat:

1. Finální seznam řešených integračních procesů a případů a jejich zařazení do určité skupiny pro detailní analýzu a návrh - viz kapitola 9 Integrační proces a integrační případ.
2. Popis integračních cest řešení s přehledem integračních procesů, které jimi budou protékat.
3. Architekturu celého řešení ve vazbě na integrační platformu.
4. Koncepční specifikaci klíčových sdílených modulů.
5. Specifikaci podnikových procesů podporovaných integrační platformou.

Výstup globální analýzy a návrhu je mimo jiné rozhodující pro nákup předpřipravených komponent (např. konektorů, či doplňků integrační platformy) či pro proprietární vývoj.

Podle aktuálního poznání potřeb projektu (např. se identifikují nové aplikace potřebné k integraci) dochází k detailní specifikaci a aktualizaci smluvních požadavků.

4.5 Detailní analýza a návrh

V této etapě projektu se analyzují požadavky na jednotlivé integrační případy na úrovni jednotlivých položek předávaných entit, definují se požadované transformace dat a specifikují se rozhraní integrovaných aplikací. Popis integračních případů je podrobně popsán v kapitole 9 Integrační proces a integrační případ. Dále se analyzují požadavky na integrační platformu jako celek (např. monitorování, administrace, zabezpečení, výkonnost).

Výsledkem této fáze je podrobný popis integračních případů. Tím se uzavírá fáze analýzy postupující od podnikových procesů přes integrační procesy až k integračním případům. Dle této analýzy se nyní modelují řešení pro integrační případy. Zde dochází k podstatné změně přístupu mezi analýzou a návrhem. Zatímco analýza byla orientována procesně, pak návrh je orientován podle integračních komponent. K posunu dochází proto, že vývojové týmy jsou rozdělovány podle integračních technologií (tedy např. vývoj ODS, nastavení ETL, modelování procesů na integračním serveru) a návrh orientovaný podle procesů průřezově přes všechny integrační komponenty by byl pro týmy nepřehledný a neefektivní (např. vývoj sdílených komponent).

Důrazně však upozorňuji, že je neustále dbát na provázanost integračních komponent, zejména průřezově kontrolovat, zda integrační procesy skutečně protékají skrze všechny komponenty správně (tuto kontrolu zpravidla provádí architekt integrace a

architektem podnikových procesů). Za tímto účelem je třeba ve specifikacích neustále udržovat vazbu na integrační případy, které se k navrhované funkcionalitě vztahují. Bez toho je i pozdější rozvoj a údržba komponent velmi obtížná.

S rostoucím počtem integrovaných aplikací je nutno dbát na dobře strukturovanou a především provázanou dokumentaci, kterou je možno jednoduše využít při analýze závislostí mezi aplikacemi (zejména při změnovém řízení) a dopadů požadovaných změn.

Popis integračních případů, resp. procesů musí být verifikován oproti funkcionalitě (např. use case modelům) integrovaných aplikací (tzv. integračními body, u kterých se definují předávaná data a jejich struktura). Specifikaci rozhraní aplikace na straně aplikace musí dodat vlastník aplikace (který ve většině případů zajišťuje i vývoj na straně aplikace).

Výstupem detailního návrhu je tedy specifikace pro vývoj (podle rozsahu projektu):

1. Specifikace rozhraní integrovaných aplikací.
2. Specifikace úprav a vývoje integračních komponent, dle rozsahu řešení:
    a. integračního serveru (rozhraní BPI, modely podnikových procesů (transformační a korelační postupy), metadata, formát kanonických zpráv, ...),
    b. ODS/DWH (datový model, rozhraní databáze, interní funkcionalita),
    c. ETL (transformační postupy, časový plán, mapování dat),
    d. XQuery Gateway (metadata, rozhraní v podobě návrhu náhledů),
    e. EIP (složení obrazovek z portletů, rozhraní portálu na aplikace i další integrační komponenty, návrh portálové databáze a další funkcionalita portálového frameworku),
    f. dalších sdílených komponent využitých pro integraci.
3. Rozšířené funkcionality integračních komponent (např. odlišná administrační konzole pro monitorování integračních procesů, nadstandardní zabezpečení,...).
4. Řešení integračních případů jinými způsoby (např. přímou komunikací)

Používanými integrační postupy se zabývá kapitola 10 Integrační postupy.

Po zpracování detailního návrhu se opět přistupuje k posouzení projektu z hlediska očekávaných nákladů a přínosů a podle rozhodnutí se může opravit rozsah projektu. V této fázi se zpravidla zkouší POC2 pro ověření funkcionality a výkonnosti klíčových částí integrace (např. na vybraném integračním procesu).

4.6 Implementace

Detailní návrh je zadávacím dokumentem pro vývoj potřebných komponent a jejich vzájemnou spolupráci. Standardní části hlavních integračních komponent jsou popsány v kapitole 12 Modely integračních nástrojů. Vývoj probíhá (zpravidla paralelně) v těchto oblastech:

1. vývoj či úpravy sdílených komponent integrační platformy,
2. vývoj či úpravy rozhraní integrovaných aplikací,
3. vývoj či úpravy rozhraní integrační platformy pro navržené integrační případy.

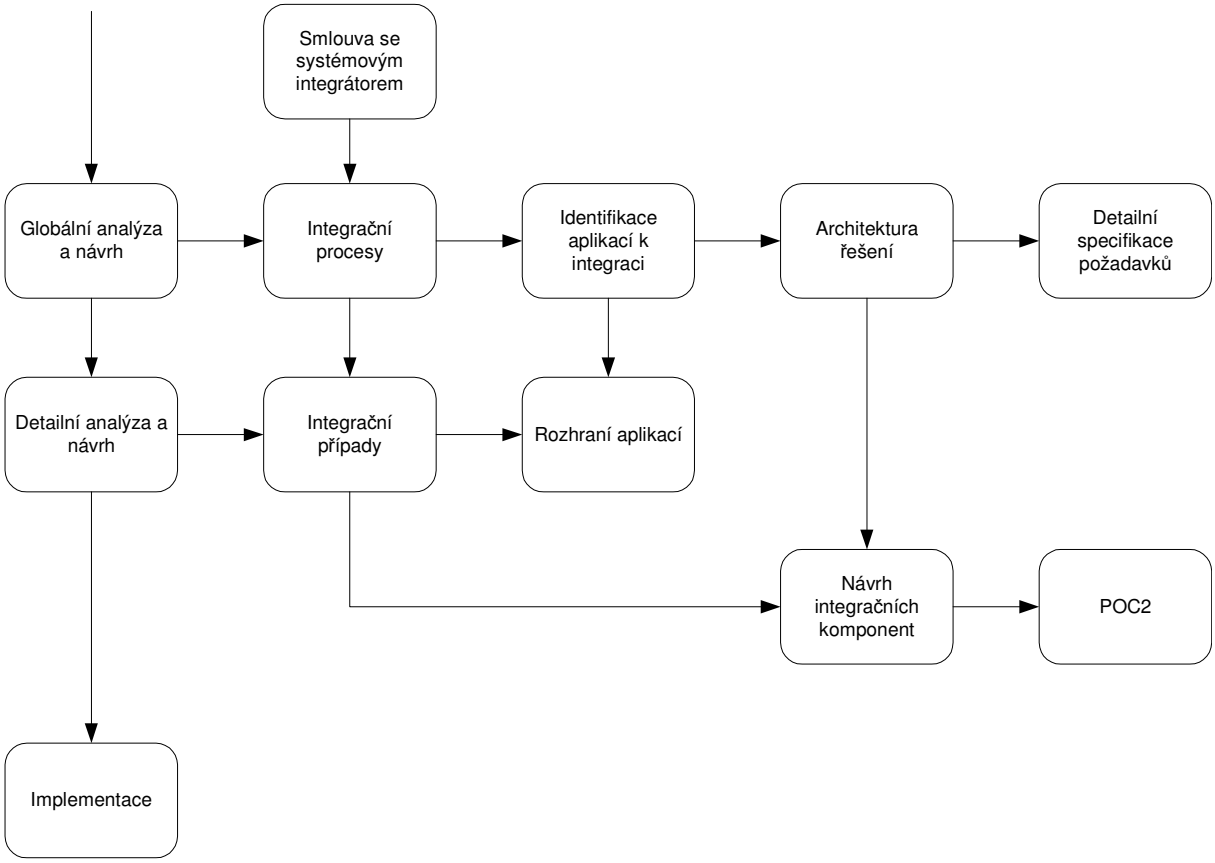
Během vývoje je třeba neustále hlídat vzájemné vztahy mezi vyvíjenými komponentami - ty by měl samozřejmě zohledňovat detailní projektový plán. Příkladem může být vývoj rozhraní aplikace (nebo např. ODS), na které čekají ostatní vývojové týmy, neboť pro svůj vývoj se potřebují navázat na fyzicky existující rozhraní.

Vývojovým postupům se v této práci nebudeme explicitně věnovat, neboť jsou závislé na zvolené integrační technologii a produktu a jsou dostatečně dokumentované v příslušných informačních zdrojích.

Součástí implementace je rovněž testování, které v případě integračních projektů musí být opravdu důkladné, neboť chyba v propojení systémů může vést ke kritickým dopadům na celý podnik. Testování probíhá pomocí tzv. testovacích scénářů (test cases),

které simulují všechny možné situace daného integračního případu. Testování probíhá zpravidla podle jednotlivých integračních případů, kde se nejdříve testuje funkcionalita rozhraní na straně integrační platformy, poté rozhraní na straně aplikací, a teprve po odladění obou rozhraní se testuje propojení obou rozhraní jako celého integračního případu. A po otestování integračních případů se testuje celý integrační proces.

Testovací příklady by měly vznikat již ve fázi detailní analýzy a návrhu, resp. být odvozeny od popisu funkcionality (např. use case modelů při aplikaci UML). O testování pojednává kapitola 15 Testování.



obrázek 11 - Fáze GAN a DAN v integračním projektu (zdroj: autor)

4.7 Zavádění

Proces zavádění patří v integračních projektech mezi poměrně složité. Jak již bylo zmíněno v kapitole 2.2 Typy projektů, je třeba z hlediska zavádění vždy rozhodnout, zda se bude jednat o skokový přechod (boom), nebo zda po určitou dobu budou udržovány oba systémy (nový i starý) souběžně. Toto rozhodnutí má výrazný dopad již od fáze detailní analýzy, neboť na něj musí být aplikace i integrační platforma připraveny (např. se musí řešit situace, jak se má aplikace zachovat, když objednávka přijde současně přes nové i staré rozhraní). Důležité je si uvědomit, že ošetření souběhu systémů znamená v důsledku velké množství programového kódu navíc a v okamžiku ukončení souběhu je většinou nutné provést další upgrade aplikací odstraňující nadbytečný kód (i tento upgrade může způsobit výrazné škody).

Proces zavádění začíná instalací integračních komponent a upgradů stávajících rozhraní aplikací. Postup je následující:

1. Upgrade zdrojové, či volané aplikace (data či služby jsou do provedení následných kroků tzv. mrtvá, tedy nevyužívaná).

2. Upgrade integrační platformy (data či služby jsou akceptována ze zdrojové aplikace a předána na rozhraní cílové aplikace, která data nepřevezme).
3. Upgrade cílové, či volající aplikace.

Tento postup lze však uplatnit jen v některých případech, v ostatních případech je nutno provést všechny kroky naráz (zejména tehdy, kdy upgrade rozhraní aplikace znamená odebrání některých dat, případně jsou data poskytována v odlišné struktuře), nebo provést upgrade v několika fázích (členěných dle předávaných dat). Rozhodnutí o zvoleném přístupu může být rovněž ovlivněno tím, zda je možné upgradovaný systém po určitou dobu odstavit (zpravidla v nočních hodinách), nebo zda je možné provést tzv. horký upgrade (bez zastavení systému za definovaných podmínek).

Pokud zavádíme i integrační komponenty typu ODS a DWH, které samy uchovávají data z aplikací, pak se řeší i fáze nalití dat do těchto komponent. V praxi to znamená, že řádově několik dní i týdnů se budou data ze zdrojové aplikace replikovat do této komponenty. Pro zdrojovou aplikaci to znamená dočasnou nadstandardní zátěž (zpravidla pouze po dobu vylití dat do DSA). Doba nalévání dat se před samotným spuštěním velmi těžko odhaduje a je nutné na ni mít v projektovém plánu vyčleněnu rezervu.

Nalévání dat je podstatnou součástí migračního plán, který určuje, jakým způsobem budou do upgradovaných či nových aplikací převzata data z původních aplikací (jedná se o změny struktur, slučování a rozdělování dat, nahrazování dat apod.). Řešení této problematiky rovněž výrazně navyšuje náklady na dočasný souběžný provoz starých a nových aplikací. Při aplikaci migračního plánu se provádí i čištění produkčních dat, neboť s otevřením rozhraní zdrojových aplikací se zvyšují požadavky na čistotu dat.

Fáze zavádění se ukončuje produkčním testem, který spustí integrační procesy v produkčním prostředí a ještě jednou se všechny kritické požadavky znovu kontrolují. Pokud je to možné, tak před ukočením produkčního testu by se integrační procesy neměly využívat pro potřeby podniku. V případě úspěšného výsledku produkčního testu se odpojí (VY) původní řešení a začíná běžný provoz.

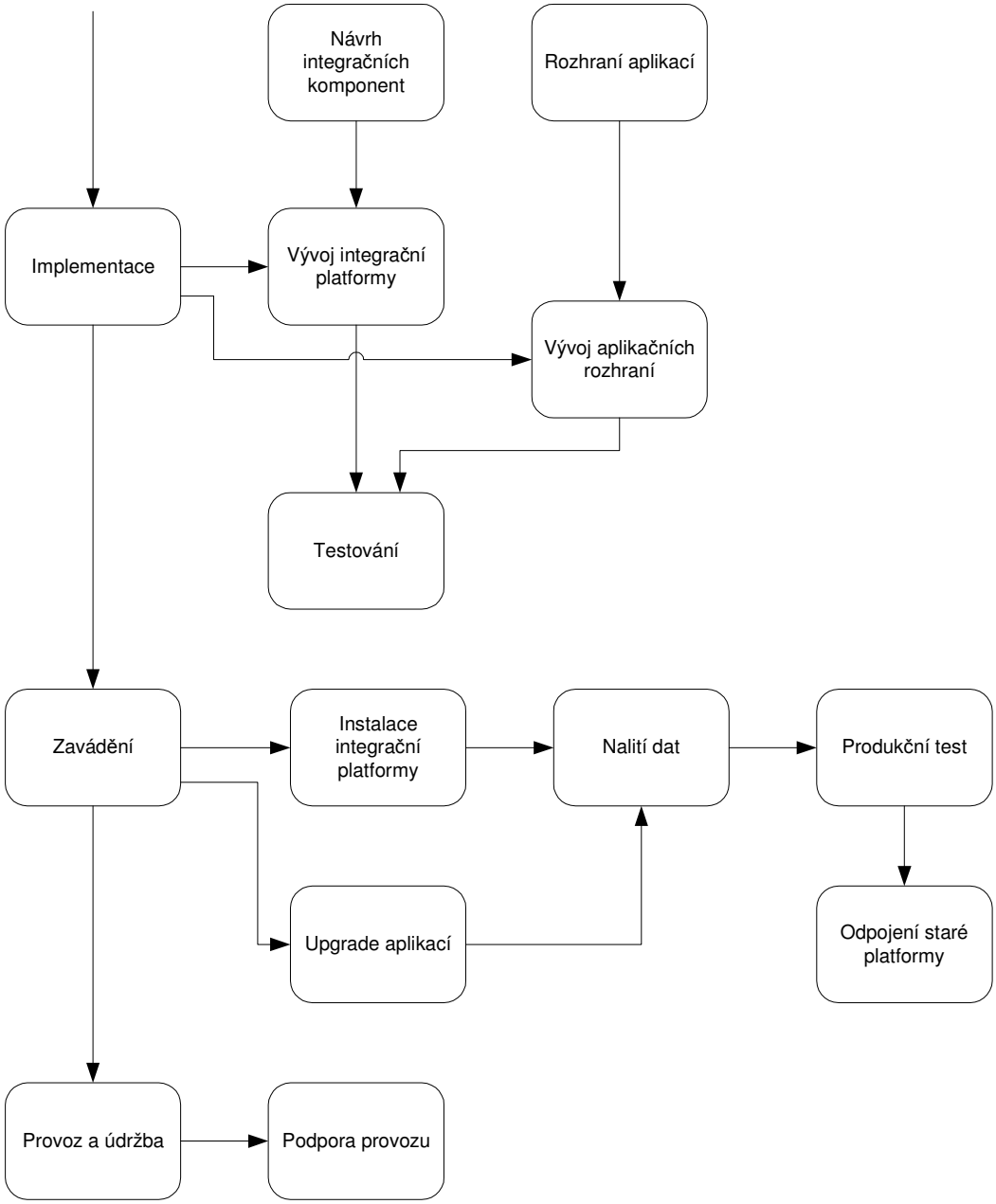
4.8 Provoz a údržba

Provoz a údržda integrovaného PIS je podobná s provozování jednotlivých aplikací. Tím, že integrační projekty nejsou zpravidla projekty zaměřené přímo na cílové uživatele (avšak i takové případy mohou nastat, kdy např. modul na integrační platformě měří KPI, ale o tom uvažujme jako o určité nadstavbě integrační platformy), jsou provoz a zejména požadované změny integrační platformy (ve smyslu integračních případů) řízeny nepřímo prostřednictvím integrovaných aplikací.

Integrace je nekončící proces, který oživují další impulsy (viz také [BUSSLER, 2003]):

1. integrace dalších aplikací,
2. změna integračních cest u vybraných integračních případů (přechod na efektivnější cestu - viz kapitola 11 Model komplexní integrované architektury PIS),
3. rozvoj integrační platformy (přechod na novější verzi, nebo jiný produkt, nové verze integračních standardů),
4. upgrade již integrovaných aplikací (a tím i rozhraní),
5. implementace podpory dalších podnikových procesů,
6. úpravy stávajících podnikových procesů (např. nové obchodní produkty, optimalizační úpravy),
7. opravy chyb.

Integrační platforma se s rostoucím rozsahem podporovaných podnikových procesů a integrovaných aplikací stává nejkritičtější aplikací celého podnikového IS. Pro provoz integrační platformy je zajišťována podpora 24x7 (24 hodin denně, 7 dní v týdnu) a SLA na opravu chyb jsou velmi přísná (déletrvající chyba může zablokovat chod celého podniku).



obrázek 12 - Fáze IM, ZA a PU v integračním projektu (zdroj: autor)

4.9 Ostatní projektové procesy

Po celou dobu existence projektu běží tyto dva procesy:

- **Projektové řízení**, které průběžně monitoruje stav projektu v konfrontaci s projektovým plánem a v případě potřeby realizuje opatření pro korekci tohoto stavu. Řízením projektů se zabývá např. [JALOTE, 2002], [FIALA, 2004], [DUNCAN, 1996], [TAYLOR, 2004]. Přehledně mapuje vztah mezi projektovými fázemi metodiky MMDIS a projektovým řízením v podobě stádií postupu projektu (přípravou , naplánováním, řízením postupu včetně ošetření změn a ukončením projektu) [ŘEPA, 2006b].
- **Řízení změn** (Change management), které detekuje dodatečné požadavky nad rámec stávajícího řešení, vyhodnocuje jejich dopad a v případě jejich akceptace je předává projektovému řízení k zapracování do projektového plánu. Zásadami řízení změn se detailně zabývá např. [HASS, 2002].

5. Dimenze integračního projektu

V souladu s mateřskou metodikou MMDIS se v této kapitole budeme věnovat průřezovým dimenzím projektu, které se prolínají všemi fázemi projektu. K dimenzím MMDIS je doplněna navíc dimenze bezpečnostní.

Podle metodiky MMDIS, kterou opět využijeme, se zkoumají tyto dimenze (převzato z [VOŘÍŠEK, 1997]):

1. Obsahové
    a. Informační, datová (INF) - zabývá se typy informací, kterých je třeba při jednotlivých podnikových aktivitách, obsahem a strukturou datové základny podniku a jejím fyzickým uložením.
    b. Procesní, funkční (PRO) - zabývá se podnikovými procesy a možnostmi jejich podpory funkcemi podnikového IS.
    c. Ekonomická, finanční (EKO) - zkoumá finanční náklady tvorby a provozu informačního systému a přínosy podniku z jeho užití.
    d. Organizační, legislativní (ORG) - zkoumá soulad zaváděného systému s legislativou, standardy a normami a vliv nového systému na organizační strukturu podniku.
    e. Pracovní, sociální, etická (PRA) - určuje potřebnou strukturu pracovníků podniku pro stav po zavedení nového systému a analyzuje sociální a etické důsledky přechodu na nový systém.
    f. Softwarová (SW) - definuje architekturu programového systému, resp. jeho jednotlivé softwarové komponenty a vztah mezi nimi.
    g. Hardwarová (HW) - definuje hardwarovou architekturu systému, resp. jeho hardwarové komponenty a vztah mezi nimi.
2. Metodicko-organizační
    a. Metodická (MET) - určuje metody a s nimi související techniky a nástroje používané v jednotlivých fázích vývoje informačního systému.
    b. Dokumentační (DOK) - určuje, jaké dokumenty vznikají v jednotlivých fázích vývoje systému, a jak na sebe navazují.
    c. Manažerská (MNG) - určuje postup při řešení jednotlivých fází vývoje informačního systému a dále pravidla a organizaci tvorby a provozu systému

[VOŘÍŠEK, 1997] dále popisuje vzájemnou provázanost popsaných obsahových dimenzí, které se zde však věnovat nebudeme (tato provázanost je platná i pro popisovanou metodiku). My se budeme věnovat detailnímu zkoumání obsahu jednotlivých dimenzí v pojetí integračního projektu s cílem najít specifické vlastnosti tohoto typu projektů. Předmětem projektu bude pro toto zkoumání nasazení kompletní integrační platformy (především typu integračního serveru, pokud nebude řečeno jinak) do prostředí PIS.

V užším pojetí můžeme tento projekt chápat jako nasazení specifické aplikace mezi další stávající podnikové aplikace, v širším pojetí jako propojení podnikových aplikací coby samostatných prvků do jednoho velikého systému. Obě pojetí jsou správná a je třeba je mít neustále na paměti, neboť zkoumání dimenzí projektu IS se týká obou z nich.

5.1 Dimenze INF

Datová základna integrační platformy je odlišná od klasických projektů implementace vybrané aplikace. Tato odlišnost vyplývá ze samé koncepce integrace - integrační platforma není primárně určena pro ukládání, či přímé zpracování dat. Naopak slouží pro předávání dat mezi integrovanými systémy.

V širším pojetí datové základny jako součásti komplexního podnikového informačního systému zahrnujícího všechny aplikace i integrační platformu můžeme datovou základnu

zachytit v konkrétní instanci modelu komplexní integrované architektury PIS (viz kapitola 11 Model komplexní integrované architektury PIS), který včetně doprovodného textu popisuje tyto integrační prvky:

- integrované aplikace,
- integrační platformu (komponenty),
- rozhraní mezi aplikacemi a integrační platformou,
- hlavní směr toku dat přes rozhraní (v případě volání služeb aplikací se dle potřeby znázorňuje směr volání služby),
- datové entity a jejich primární zdroj,
- distribuci datových entit přes rozhraní a integrační platformu.

V užším pojetí integrace, kdy datovou základnu zkoumáme pouze u integrační platformy, nalézáme její hlavní části v:

1. ODS/DWH - datovou základnu tvoří entity datového skladu, zpravidla se jedná o složité, univerzálně postavené datové struktury.
2. Ostatních integračních komponentách ve formě metadat, pomocných číselníků a cache (např. message store v integračním serveru).

Během analýzy a návrhu vzniká komunikační schéma i detailní popis datové základny integrační platformy. Mimoto vznikají popisy rozhraní mezi aplikacemi na úrovni datových položek (viz kapitola 9.3 Analýza integračního případu). Z dokumentace musí být dohledatelné tyto informace o datových entitách:

- identifikace,
- vlastník,
- primární zdroj,
- distribuce dalším aplikacím,
- transformace během distribuce,
- atributy,
- vlastnosti (doba aktualizace, přístupová práva, charakter dat (např. důvěrné)).

5.2 Dimenze PRO

V širším pojetí integrační platformy vnímáme tuto dimenzi jako souhrn podnikových procesů podporovaných integrační platformou. Tato podpora bude v procesních diagramech příslušným způsobem znázorněna, aby bylo jasné, které aktivity procesu jsou zautomatizovány pomocí integrační platformy a navázaných aplikací (ty označujeme jako integrační procesy). Popis podnikových procesů by měl rovněž zahrnovat alternativní cestu procesu pro případ kolapsu integrační platformy či jiných vyjímečných stavů.

V užším pohledu na integrační platformu v této dimenzi budeme zkoumat zejména služby (funkce) poskytované API integrovaných aplikací a jejich propojení v podobě integračních procesů na úrovni integrační platformy. Integrace na datové úrovni podporuje podnikové procesy přenášením dat mezi systémy, neumožňuje však výraznější podporu na funkční úrovni (neuvažujme nyní technologii databázových triggerů, které omezenou podporu umožňují).

Naopak integrace na funkční úrovni umožňuje více možností podpory automatizovaných procesů. Platformy typu integrační server nabízejí vizuální modelování a řízení částí podporovaných podnikových procesů. Na celý PIS můžeme nahlížet jako na skupinu poskytovaných služeb a takto může být i popsán - viz kapitola 17.1.8 SOA, kde integrační platforma podle zavedených podnikových procesů řídí volání jednotlivých služeb.

Projektová dokumentace musí obsahovat popis podporovaných částí podnikových procesů. Implementovaný způsob podpory procesů integrační platformou je detailně zdokumentován v popisu integračních případů (viz kapitola 9.2 Integrační proces), zejména v

tzv. workflow části. Komunikační schéma v této dimenzi popisuje směr volání služby (resp. která aplikace iniciovala spuštění služby).

5.3 Dimenze EKO

Posuzování předpokládaných přínosů a nákladů integrační platformy jako celku jsme se věnuje samostatná kapitola 7 Business case. Z hlediska posuzování návratnosti je však nutné posuzovat v podstatě každý integrační případ. V praxi se například můžeme setkat s případem integrace mezi dvěma aplikacemi, kde bude třeba provést náročnou analýzu vzhledem k předpokládaným transformacím, vyřešit úpravu API obou aplikací typu black-box atd., což povede k velkým nákladům na vybudování tohoto rozhraní. Reálné užití tohoto rozhraní však proběhne řádově 4 krát do roka a jedná se o velmi malé množství dat. V takovém případě je zpravidla výhodnější realizovat přenos dat manuálně. Podobných případů je celá řada. Není tedy výhodné se snažit slepě pro všechny integrační případy nalézt řešení pomocí integrační platformy za každou cenu.

Bolestivým místem posuzování jednotlivých integračních případů je řešení vyjímečných stavů. Není zpravidla problém vcelku přesně vyčíslit očekávanou návratnost automatické podpory daného integračního případu. Problém může nastat v okamžiku, kdy je třeba zanalyzovat a vyřešit všechny možné stavy (nikoli pouze ty optimální, očekávané), které sice mohou nastat, ale pouze vyjímečně. Těchto stavů je řádově několikrát více než těch standardních a jejich řešení neúměrně navyšuje cenu celého řešení. Opět je třeba stanovit hranici, které stavy lze efektivně řešit automaticky a které je efektivnější řešit manuálně.

5.4 Dimenze ORG

V případě integračních projektů se můžeme s legislativou setkat v těchto případech:

1. Přenášená data patří do některé z kategorií chráněných právním systémem (např. osobní údaje) a je třeba je podle toho zabezpečit (jak při přenosu, tak i při dočasném uložení).
2. U aplikace je třeba provést úpravu či vytvořit API aplikace, avšak zásah do aplikace je chráněn autorským zákonem, či smluvním ujednáním. Tuto situaci je třeba řešit individuálně s dodavatelem aplikace, případně zvolit jinou aplikaci nebo jiný způsob integrace.

Z hlediska organizační struktury dochází k vytvoření nových pracovních pozic okolo integrační platformy (viz kapitola 6 Projektový tým) a naopak ke změně náplně či zániku pracovních pozic realizujících manuálně komunikaci mezi aplikacemi. Integrační projekt často slouží jako iniciátor změn v podnikových procesech, které však nelze obecně specifikovat a je nutné je analyzovat individuálně podle očekávaných přínosů projektu.

5.5 Dimenze PRA

Nasazení integrační platformy vede k automatizaci částí podnikových procesů, a tím k nahrazení manuálních aktivit zaměstnanců. V optimálním případě umožňuje tato změna vykonávat pracovníkům pracovní činnost efektivněji a převést je na jinou práci. Jiným výsledkem však může být i ztráta pracovního místa, což má přirozeně negativní dopad na zaměstnance. Během nasazování integrační platformy, a zejména při vytváření rozhraní mezi aplikacemi, jsou však zpravidla znalosti těchto pracovníků klíčové pro úspěch projektu, neboť znají nejen základní, optimální proces předávání dat mezi aplikacemi, ale znají i vyjímečné stavy procesu. Z tohoto důvodu je třeba s klíčovými pracovníky pracovat velmi citlivě a správně je motivovat.

5.6 Dimenze SW

Softwarová architektura integrační platformy je popsána v kapitole 12 Modely integračních nástrojů, a proto ji zde již nebudeme detailně rozebírat. Komplexní pohled na provázanost aplikací a integrační platformy poskytuje model popsaný v kapitole 11 Model komplexní integrované architektury PIS.

5.7 Dimenze HW

Hardwarová dimenze v integračních projektech má dvě důležité složky:

- Komunikační infrastrukturu pro komunikaci aplikací - v žádném případě by neměla být limitujícím faktorem výsledného řešení. Doporučené (dle [HERRERA, 2003]) dimenzování propustnosti počítačové sítě by mělo být o cca 30-50 procent vyšší než bude standardní průtok dat. K tomu by měla být infrastruktura vybavena bezpečnostními prvky (např. firewally) a výkonnostními prvky (load balancer).
- Počítačové servery pro řízení a administraci řešení - hraje podstatnou roli v topologii hvězdice (integrační hub, ODS, DWH, XQuery Gateway, EIP), kde se velmi snadno může stát úzkým hrdlem celého řešení. Odhadnout parametry potřebné konfigurace serverů v době podpisu smlouvy je více než sázkou do loterie, proto se do smlouvy doplňují nejrůznější předpoklady, z jakých se při odhadu vycházelo. Je osvědčenou metodou požádat dodavatele integrační platformy o expertní odhad, neboť on jediný má nejvíce statistických dat a zkušeností s danou platformou.

Tato řešení jsou navrhovaná jako vysoce dostupná (HA - High Availability) řešení, což znamená operativní zálohování (duplikace) všech prvků (databáze, integrační komponenta, síť), které mohou selhat. To znamená, že v okamžiku selhání daného prvku jej během velmi krátké doby nahradí záložní prvek a převezme všechny rozpracované úlohy. Pro tyto prvky je vyžadována podpora 24x7.

Vedle provozního prostředí existuje i vývojové a testovací prostředí, které by mělo být obrazem (zpravidla s nižším výkonem) provozního prostředí. Tato prostředí jsou fyzicky oddělená a nepropojená z důvodu bezpečnosti.

5.8 Dimenze MET

Jak již bylo zmíněno v předchozích kapitolách, neexistuje obecně známá metodika specifická pro integrační projekty, každý integrátor používá svou metodiku. Pro zákazníka je důležité porozumět jednotlivým fázím a jejich výstupům, aby se v projektu dostatečně orientoval a mohl jej průběžně vyhodnocovat. V kapitole 2.3 Obecné úrovně integrace jsme se zmínili o metodologické integraci, zajišťující provázání metodiky implementace integrační platformy s metodikami integrovaných aplikací (např. v otázce upgrade aplikací), proto je dobré se zaměřit i na tuto oblast metodiky integrátora.

5.9 Dimenze DOK

Dokumentace vznikající během integračního projektu se příliš neodlišuje od standardních projektů. Přestože některé dokumenty již byly popsány v předchozích kapitolách, provedeme zde jejich stručné shrnutí dle výstupů jednotlivých fází projektu:

1. GST, IST
    a. Informační strategie obsahující integrační strategii
    b. Business case
    c. Specifikace projektu obsahující projektový plán a specifikace požadavků a jejich akceptační kritéria
2. US
    a. Úvodní studie

   b. Smlouva s integrátorem a dodavatelem integrační platformy (včetně garance parametrů platformy)
3. GAN
   a. Specifikace s upřesněnými požadavky
   b. Globální analýza a návrh obsahující integrační procesy, architekturu řešení a dopad na integrované aplikace
4. DAN
   a. Analýza integračních případů
   b. Analýza a návrh rozhraní aplikací
   c. Návrh implementace integračních komponent (technická specifikace)
   d. Zápisy z jednání
   e. Migrační plán
5. IM
   a. Testovací příklady
   b. Testovací protokol
   c. Zdrojové kódy
   d. Instalační balíčky
   e. Update dokumentace k aplikacím
   f. Technická dokumentace integrační platformy
   g. Technická dokumentace k rozhraní
   h. Provozní dokumentace (administrační (obsahující popis instalace, zálohování, správu systému, havarijní plány atd.) a uživatelská příručka) integrační platformy
6. ZA
   a. Instalační protokol
   b. Akceptační protokol (s výsledkem bezpečnostního auditu)
7. PU
   a. Provozní protokoly
   b. Změnové požadavky

5.10 Dimenze MNG

Řízení integračního projektu se podstatně neodlišuje od řízení standardních projektů.

5.11 Dimenze SEC (doplněná)

Nad rámec metodiky MMDIS doporučuji přidat jednu obsahovou dimenzi - bezpečnostní (SECurity), která pokrývá zabezpečení celého systému. Zabezpečení systému znamená zajištění následujících tří vlastností systému (viz také [BUSSLER, 2003]):

1. Důvěrnost - k datům a poskytovaným službám má přístup pouze oprávněná osoba na základě přiřazených práv (souvisí s identifikací, autentikací a autorizací komunikujících stran).
2. Integrita - data a poskytované služby nemohou být narušeny neoprávněnou stranou bez vědomí dotčených stran.
3. Dostupnost - data a poskytované služby jsou k dispozici oprávněným uživatelům bez ohledu na činnost třetích stran.

Bezpečnost se týká jak samotné integrační platformy, tak přenášených dat. Hranice řešené oblasti tedy vymezují rozhraní integrovaných aplikací. Přes tato rozhraní se však předávají vybrané údaje mající vztah k bezpečnosti, jako např. identifikace uživatelů a jejich přístupová práva, nebo metadata k datům obsahující informaci s žádostí o zabezpečení dat na úrovni důvěrné.

Tato dimenze se prolíná celým životním cyklem projektu, kde ve fázi analýzy se zjišťují konkrétní požadavky na zabezpečení a možnosti řešení těchto požadavků. Každý

podnik by měl mít zpracovanou bezpečnostní strategii včetně metodických pokynů obsahujících konkrétní:

- klasifikaci dat (např. důvěrné) a požadavky na jejich zabezpečení
- klasifikaci aplikací (např. kritická) a požadavky na jejich zabezpečení

Každý vlastník (ve smyslu aplikace) dat by pak měl mít tato data oklasifikována. Na základě těchto informací pak s daty bude nakládat integrační platforma.

Během návrhu se definují způsoby zabezpečení jak integrační platformy, tak přenášených dat. V době implementace pak vznikají podklady pro bezpečnostní audit, které popisují implementované bezpečnostní prvky.

Součástí těchto podkladů jsou i základní administrátorské postupy (např. zálohování, monitorování) a řešení vyjímečných situací (tzv. havarijní plány). Během předávání systému probíhá bezpečnostní audit, jehož cílem je ověřit veškeré bezpečnostní prvky a najít slabá místa systému. Bezpečnostní audit probíhá na několika úrovních - např. hardware, operační systém, počítačová síť, záložní systém, aplikační balíky. Během provozu se monitorují bezpečnostní prvky (např. auditní logy) a vyhodnocují se.

6. Projektový tým

V této kapitole popíšeme základní organizační strukturu projektového týmu v integračním projektu.

Projektový tým se skládá z mnoha rolí, v kterých působí konkrétní lidé. Každá role je definována zodpovědností za určitou část projektu a pravomocí dělat v této oblasti závazná rozhodnutí. Podle rozsahu projektu může jedna osoba vykonávat i více rolí současně.

Dále popsanou strukturu nelze brát jako fixní - uspořádání týmu se liší zejména v závislosti na typu projektu a dle obchodních vztahů mezi zákazníkem, integrátorem a dodavatelem integrační platformy. Rovněž tak názvy rolí se liší vždy podle konkrétní užité metodiky.

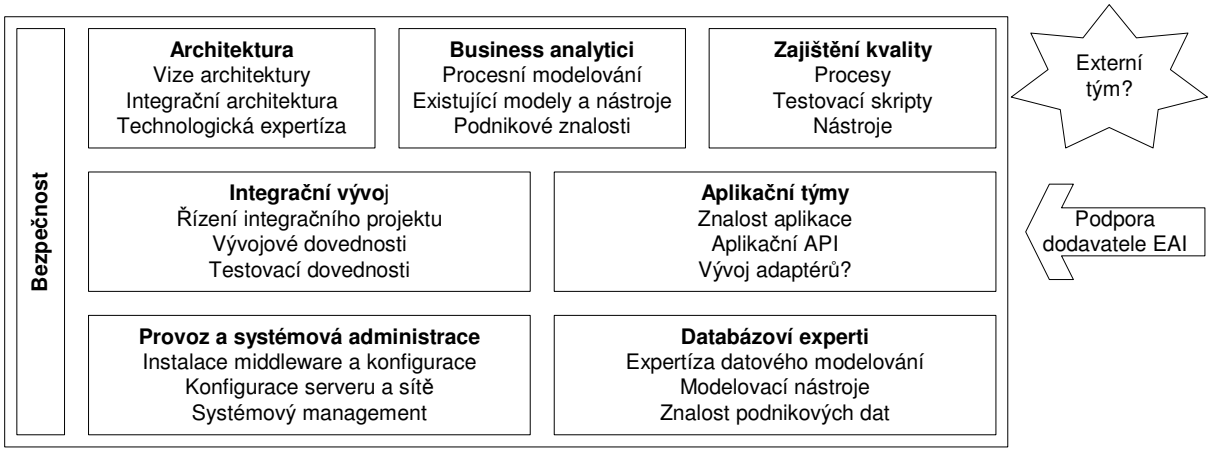
6.1 Integrační týmy v informačních zdrojích

Do tabulky 3 - Přehled integračních týmů v informačních zdrojích (zdroj: autor) jsem shrnul struktury integračních týmů, které jsem nalezl v informačních zdrojích. Jelikož ani jednu z uvedených struktur nepovažuji za kompletní, popíši integrační tým ve vlastní struktuře, v jaké jsem se s ním setkal v praxi na integračních projektech.

Identifikace informačního zdroje	**Pokrytá oblast**
[BIEBERSTEIN, 2005]	Popisuje tým pro SOA projekty s těmito rolemi: IT projektový manažer, business analytik, architekt, vývojář, specialista na bezpečnost, systémový a databázový administrátor, instalátor služeb, tester integračních služeb, vývojář nástrojů, držitel know-how, SOA projektový manažer, SOA systémový administrátor, SOA architekt, návrhář služeb, návrhář procesních toků, vývojář služeb, integrační specialista, tester interoperability, administrátor UDDI, návrhář UDDI, správce služeb.
[BUCHALCEOVÁ, 2005]	Popisuje projektový tým pro objektově orientovaný vývoj nového obecného software vlastními silami. o Celopodnikové role SW inženýrské o celopodnikový inženýr pro znovupoužitelnost o tvůrce vzorů o správce databáze o celopodnikový architekt o Celopodnikové řídící role o informační manažer o Role zájmových skupin o sponzor o vedení organizace o uživatel o Doménové role o specialista na doménu o vlastník služby o vlastník procesu o SW inženýrské role v projektu o hlavní programátor o vlastník třídy o specialista na požadavky o specialista na UI o analytik o návrhář o programátor

	○ vývojář ○ tester ○ analytik testů ○ manažer testů ○ návrhář testů ○ dokumentátor o Řídící role v projektu ○ vedoucí projektu o Další role ○ sociolog a psycholog ○ usnadňovač (facilitator)
[HERRERA, 2003]	Definuje tyto týmy a role v EAI projektech: o Vývojový ○ Projektoví ředitelé a manažeři ○ Vývojáři ○ Lídři týmů ○ Business analytikové ○ Koncoví uživatelé ○ Aplikační specialisté o Provozní ○ Systémová administrátoři ○ Síťoví administrátoři ○ Databázoví administrátoři ○ Integrační architekt ○ Technický architekt ○ Databázový architekt o Top management ○ Obchodní lídři
[MCCOY, 2003a]	viz obrázek 13 - Týmy a role v integračním projektu dle (zdroj: [MCCOY, 2003a])

tabulka 3 - Přehled integračních týmů v informačních zdrojích (zdroj: autor)



obrázek 13 - Týmy a role v integračním projektu dle (zdroj: [MCCOY, 2003a])

6.2 Interní týmy

Následující model vychází z těchto předpokladů:

1. Cílem projektu je nasazení kompletní integrační platformy do podniku. Pokud by se jednalo o projekt, jehož cílem by byla implementace konkrétní aplikace a integrace této aplikace by byla pouze součástí tohoto projektu, do organizační struktury by byl

začleněn tým implementátora aplikace a do jednotlivých níže definovaných týmů by byl začleněn jeho příslušný zástupce.
2. Zákazník koupil integrační platformu přímo od dodavatele a najal integrátora pro vlastní implementaci. Druhým, méně častým případem může být situace, kdy zákazník najal integrátora a předal mu zodpovědnost za výběr i nákup integrační platformy. V takovém případě zákazník s dodavatelem přímo vůbec nekomunikuje a veškeré problémy platformy se řeší pouze mezi integrátorem a dodavatelem.

Obrázek 14 - Role v integračním týmu (zdroj: autor) znázorňuje základní schéma organizační struktury integračního projektového týmu. V obrázku jsou rovněž zachyceny podstatné projektové skupiny, v jejichž čele stojí vždy vedoucí týmu, který tým vede a organizuje. Identifikovány jsou tyto projektové skupiny:

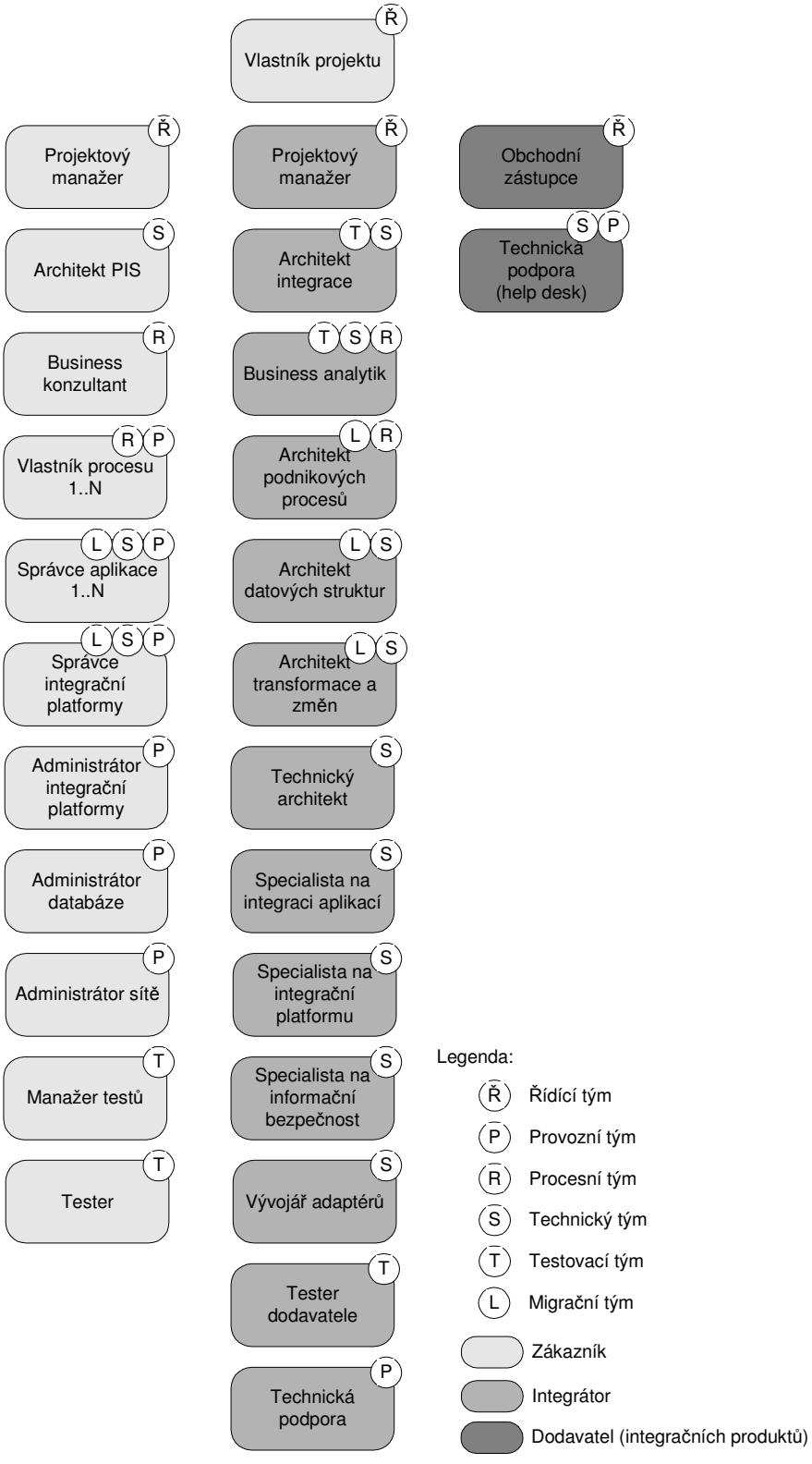
1. Řídící tým - odpovídá za chod a výsledky celého projektu, rozhoduje koncepční záležitosti.
2. Provozní tým - přebírá projekt ve fázi zavádění systému a zajišťuje vlastní provoz integrační platformy.
3. Procesní tým - v jednotlivých fázích projektu (včetně zavádění) zodpovídá za oblast podnikových procesů
4. Technický tým - v jednotlivých fázích projektu zodpovídá za vývoj a implementaci integrační platformy a vlastní integraci aplikací na technické úrovni.
5. Testovací tým - ve fázi implementace provádí testování celého řešení.
6. Migrační tým - zajišťuje přechod stávajícího podnikového IS do nového prostředí integrační platformy, zejména převod dat

6.3 Role

Role v projektovém týmu (v závorkách je uvedeno orientační mapování na role z [BUCHALCEOVÁ, 2005]):

1. Zákazník
    a. Vlastník projektu (sponzor) - zodpovídá za celý projekt na straně zákazníka, zajišťuje financování projektu.
    b. Projektový manažer (vedoucí projektu) - řídí skupinu lidí na straně zákazníka.
    c. Architekt podnikového IS (celopodnikový architekt) - zodpovídá za finální koncepci komplexního podnikového IS.
    d. Business konzultant (specialista na požadavky) - skupina lidí majících přehled o provázanosti podnikových procesů a aplikací a o požadavcích businessu, kteří tvoří detailní vizi kooperace implementovaných podnikových procesů a integrovaných aplikací.
    e. Vlastník procesu (vlastník procesu) - zástupci znalí dotčených podnikových procesů, s pravomocí rozhodovat o podpoře a úpravě těchto procesů. Často podporováni skupinami uživatelů těchto procesů.
    f. Správce aplikace (specialista na doménu, vlastník služby) - zástupci integrovaných, či jinak dotčených aplikací, s možností rozhodovat o způsobu integrace. Často doplněni o osoby znalé technických detailů aplikací, někdy též externí dodavatelé aplikací.
    g. Správce integrační platformy - odpovídá za chod a rozvoj integrační platformy z hlediska koncepce.
    h. Administrátor integrační platformy - zajišťuje chod integrační platformy.
    i. Administrátor databáze - zajišťuje chod databáze integrační platformy, často v návaznosti na ODS.
    j. Administrátor sítě - zajišťuje síťové konfigurace a chod síťového propojení integrační platformy s aplikacemi.
    k. Manažer testů (manažer testů) - plánuje běh testování, koordinuje spolupráci testerů a vývojářů opravující nalezené chyby.

    l. Tester (tester, analytik testů) - provádí testování celého řešení.
2. Integrátor
    a. Projektový manažer - řídí skupinu lidí na straně integrátora.
    b. Architekt integrace (hlavní programátor) - zodpovídá za integraci na všech úrovních.
    c. Business analytik (návrhář testů) - transformuje detailní vizi kooperace implementovaných podnikových procesů a integrovaných aplikací do požadavků na integrační platformu.
    d. Architekt podnikových procesů (návrhář) - odpovídá za integraci podnikových procesů.
    e. Architekt datových struktur (návrhář) - odpovídá za analýzu a přípravu dat pro integraci.
    f. Architekt transformace a změn (návrhář) - připravuje a realizuje transformaci IS.
    g. Technický architekt - odpovídá za výběr, instalaci a konfiguraci hardware a software.
    h. Specialista na integraci aplikací (návrhář) - odpovídá za integraci vybraných aplikací.
    i. Specialista na integrační platformu (vývojář) - konfiguruje a doplňuje integrační platformu.
    j. Specialista na informační bezpečnost - odpovídá za zabezpečení celého řešení.
    k. Vývojář adaptérů (vývojář) - odpovídá za vývoj aplikačních adaptérů.
    l. Tester dodavatele (tester, analytik testů) - provádí testování celého řešení na straně integrátora.
    m. Technická podpora - v provozním režimu řeší problémy integrační platformy a adaptérů (tzv. First line support), v případě chyb jádra integrační platformy předává problém technické podpoře dodavatele.
3. Dodavatel
    a. Obchodní zástupce - zastupuje dodavatele integrační platformy v otázkách obchodních a poskytuje požadované informace o integrační platformě.
    b. Technická podpora - v provozním režimu řeší předané problémy integrační platformy, během implementace poskytuje technickou podporu technickému týmu.



obrázek 14 - Role v integračním týmu (zdroj: autor)

7. Business case

Business case je dokument, který je vytvářen společně obchodními a IT manažery za účelem prozkoumání vybraného business problému a identifikace předpokládaných přínosů a nákladů v případě vyřešení daného problému.

Tento dokument vzniká v době přípravy projektu, odvozuje se z globální a informační strategie a je základním dokumentem pro posuzování celého projekt. V této kapitole se budeme stručně zabývat obsahem business case dokumentu a zejména se zaměříme na společné přínosy EAI projektů. Cílem této kapitoly je zdůvodnit, proč se integrační projekty realizují, co se od nich očekává, a tím doplnit jejich charakteristiku ve vztahu k vedení podniku.

7.1 Struktura dokumentu

Struktura dokumentu business case k projektu není jednoznačně definována a každý podnik ji má odlišnou. V [WSDIS, 2004] jsem nalezl nejkomplexnější osnovu. Struktura dokumentu je obecná pro všechny typy obchodních případů. Obsah business case dokumentu definuje rámcově následující struktura (přeloženo z [WSDIS, 2004]):

1. Manažerské shrnutí (Executive Summary) - obsahuje stručný popis řešeného problému (potřeby), cíle řešení, rámec řešení, předpokládané přínosy a náklady (HLE - High Level Estimation).
2. Business potřeby (Background and Needs Assessment) - popisuje současné podnikové prostředí, identifikovanou potřebou (problém či příležitosti), cílový stav.
3. Cíle (Objectives) - definuje cíle (přínosy) projektu, pokud možno pomocí metrik (např. KPI).
4. Organizační dopady (Organizational Impacts) - identifikuje entity, které budou projektem dotčeny (jiné projekty, zaměstnanci, partneři, zákazníci, atd.).
5. Procesní dopady (Process /Procedural Impacts) - identifikuje vlivy na podnikové procesy, čímž částečně vymezuje rozsah projektu.
6. Organizace projektu (Project Management and Organization, Estimated Workplan) - popisuje hrubý projektový plán, projektový tým atd.
7. Finanční plán (Estimated Financial Plan) - kvantifikuje a porovnává očekávané přínosy a náklady v určitém časovém horizontu.

Hlavním cílem tohoto dokumentu je porovnání nákladů a předpokládaných přínosů navrhovaného projektu a dle výsledku tohoto porovnání je následně rozhodnuto, zda se bude projekt realizovat. V následujícím textu se budeme podrobně zabývat hlavními argumenty pro a proti integračním projektům.

7.2 Přínosy

Porovnání očekávaných přínosů a nákladů je považováno společně s podporou podnikových cílů (definovaných globální podnikovou strategií) za nejdůležitější informace v business case dokumentu. Jak již bylo zmíněno v kapitole 2.2 Typy projektů, s integračními projekty se málokdy setkáme jako se samostatnými projekty a není tedy nutné zpracovávat specifický business case zaměřený pouze na integraci (ale i s tím se můžeme setkat, viz 17.6.4 Projekt 4 - Budování ODS (2005 - 2006)).

Na druhou stranu se ovšem prakticky v každém jiném projektu setkáme s požadavky na integraci, jejichž náklady dosahují podstatného podílu na celkových nákladech projektu. Společnost Gartner Group ([CSC, 2001]) na základě svých průzkumů uvádí, že průměrně 40 procent z rozpočtů na oddělení IT připadá na systémovou integraci. Ve výsledku nejsou tato čísla nikterak malá, a proto se pokusíme identifikovat nejčastější přínosy EAI pro podnik.

Integrace v současném pojetí EAI přináší podniku čtyři skupiny hlavních identifikovaných přínosů :

1. Podpora podnikových procesů
    a. Podnikové procesy lze s pomocí integrační platformy automatizovat a řídit na vyšší úrovni (tradiční pojetí integrace implementuje business pravidla definující procesy přímo do rozhraní aplikací, což je činí neudržovatelnými), což v důsledku znamená snižování procesních rizik a nižší náklady.
    b. Podnikové procesy lze transparentně monitorovat a vyhodnocovat (pomocí KPI, historických záznamů a podmínek).
    c. Doba mezi startovací událostí v prvním systému a adekvátní reakcí v systému druhém je významně redukována - mezi systémy a tím i procesy jsou integrací odstraněny nebo maximálně minimalizovány časové prodlevy, dosud způsobované přechody mezi nimi. EAI umožňuje vytvořit efektivní, **událostmi řízený PIS**.
2. Redukce IT nákladů
    a. Provozní - po integraci systémů je např. možno odstranit nadbytečné datové a funkční duplicity ve více systémech, a tím uvolnit systémové zdroje pro jiné využití. Dále tím, že je ke každé aplikaci definováno pouze jedno rozhraní, skrze které komunikují všechny ostatní aplikace, je možno odstranit všechna nadbytečná rozhraní, dosud definovaná pro každou aplikaci, jejichž údržba byla velmi náročná.
    b. Vývojové - využívání společných a jednotných komponent a integračních nástrojů vede zrychlení a snížení nákladů a rizik integračních projektů.
3. Redukce chyb v systému
    a. Jednoznačně definovaná rozhraní - takřka každý přechod mezi systémy uvnitř neintegrovaného systému znamená vynucenou transformaci přenášených dat (a to prostorovou, časovou i strukturovou). Definováním přesných rozhraní je množství chyb redukováno na minimum.
    b. Unifikace informací - každá informace je vytváření jasně definovaným primárním zdrojem a distribuována dle potřeby cílovým systémům. Tím se odstraňují duplicity a nekonzistence v podnikových informacích, které mohou vznikat v případě, že zdroje informací nejsou přesně vymezeny a provázány.
4. Centrální správa integrační platformy
    a. Umožňuje rychlou detekci, následné řešení chyb systému a jejich otestování.
    b. Sdílení prvků (např. přístupových práv, transakcí) umožňuje výrazně jednodušší administraci systému a zavádí jednotný koncept pro všechny integrované aplikace.

Přínosy integračních projektů se zabývá také [THEMISTOCLEOUS, 2003], který navrhuje 5 kategorií přínosů:

- organizační (např. z lépe fungujících podnikových procesů),
- manažerské (např. dosažení návratnosti investic),
- strategické (např. zlepšení spolupráce s partnery),
- technické (např. datová, objektová a procesní integrace),
- operativní (např. snížení nákladů).

7.3 Náklady

Náklady integračních projektů můžeme rozdělit do následujících skupin:

1. Náklady integrační platformy, které zahrnují náklady vytvoří integrační platformy a její spuštění. Jedná se především o softwarové licence, hardware a náklady na úpravy (customization) a doplňující vývoj integrační platformy. [BASS, 2002] uvádí, že cca 80 procent těchto nákladů je nárazově investováno během prvních 6 měsíců.
2. Integrační náklady jsou náklady na vývoj rozhraní a spolupráci mezi aplikacemi. Tyto náklady se liší podle počtu a náročnosti vytvářených rozhraní během projektu. [BASS, 2002] uvádí, že náklady na vývoj jednoho rozhraní v EAI jsou obecně o cca

25-40 procent nižší než při point-to-point koncepci integrace. To je způsobeno využíváním společných komponent a předpřipravených prvků integrační platformy. Úspory se dosahují díky následujícím komponentám a nástrojům [BASS, 2002]:

a. business process modeler and management (BPM),
b. univerzální transformační a formátovací komponenty,
c. auditovací, logovací a debugovací nástroje,
d. Předpřipravené adaptéry.

Upřesnění úspor nákladů na jeden adaptér uvádí i [ALTMAN, 2003b], kde se dočteme, že použitím EAI nástrojů u vývoje jednoduchých rozhraní ušetříme 25 procent, u středně obtížných rozhraní 32 procent a u složitých rozhraní 43 procent nákladů. Při odhadu nákladů na vývoj integračních rozhraní jsou směrodatné následující informace (viz také [LEVINE, 2002]):

a. Počet rozhraní, resp. integračních případů.
b. Charakteristika jednotlivých rozhraní:
    - Stávající stav (dokumentace, hotový kód, atd.).
    - Dopad na stávající podnikové procesy (u výrazné změny v procesu je zpravidla náročnější analýza).
    - Připravenost integrovaných aplikací.
    - Typ rozhraní z hlediska času:
        - Permanentní - rozhraní vyvíjené pro provozní, trvalé prostředí.
        - Dočasné - rozhraní sloužící pouze po určitou dobu (typicky jen po vybranou fázi projektu), které bude odstraněno, či nahrazeno.
        - Synchronizační (multiple master) - sloužící pro synchronizaci dat mezi systémy (zpravidla jeden primární zdrojový systém a více cílových systémů).
    - Požadavky na implementaci business logiky (např. rozhodovací procesy, opakování, transakční zpracování, předání do více systémů současně, zajištění sériového zpracování atd.).
    - Náročnost transformací dat (mapování dat, změna struktury dat, podporované formáty atd.).
    - Priorita rozhraní dle významu pro podnik (implikuje např. požadavky na monitorování a ošetření chyb rozhraní):
        - kritické,
        - standardní,
        - nízké.
    - Směr toku dat:
        - obousměrný,
        - jednosměrný.
    - Předpokládaný objem dat.
c. Nutná migrace dat před zavedením jednotlivých rozhraní
    - Množství dat, resp. čas potřebný k migraci (manuální či pomocí nástroje).
    - Typ migrace:
        - statická data (např. údaje o zákaznících),
        - dynamická data (např. transakční data, která se rozlišují dle transakce na uzavřené a probíhající).

3. Provozní náklady jsou náklady na provoz a údržbu integrační platformy a jednotlivých rozhraní. [BASS, 2002] uvádí předpokládanou úsporu provozních nákladů mezi 50 a 80 procenty při přechodu z point-to-point na EAI koncepci integrační platformy. [ALTMAN, 2003b] předpokládá úsporu provozních nákladů shodnou s náklady na vývoj rozhraní s tím, že roční provozní náklady lze kalkulovat jako 25 procent nákladů na vývoj.

Vzorové srovnání nákladů mezi EAI a point-to-point koncepcí je rozpracováno v [BASS, 2002].

7.4 Rizika

Každý projekt, nejen integrační, v sobě obsahuje rizikové faktory, které potenciálně mohou vést k neúspěchu projektu, a tím k nepříznivým dopadům pro podnik. Odstraňovat či minimalizovat tato rizika je úkolem projektového manažera. Rizika integračních projektů můžeme rozdělit na tyto skupiny:

A. Projektová rizika, což jsou rizika vyplývající se samé podstaty projektu (např. onemocnění klíčových členů týmu, přírodní katastrofa apod.). Ta jsou popsána v dostupné literatuře a nebudeme se jim dále věnovat, neboť jsou nad rámec této práce.
B. Integrační rizika jsou specifická rizika plynoucí z integrační povahy projektu.
C. Podnikatelská rizika jsou specifická rizika odvozená od předmětu podnikání daného podniku ve vztahu k projektu. Může se například jednat o sankce vyplývající z legislativy v případě, že se projekt nepodaří realizovat (např. přenositelnost telefonního čísla mezi operátory je požadována zákonem a operátoři ji tedy musí podporovat externí i interní integrací svých systémů).

Z integračních rizik uvádím následující příklady, s kterými jsem se setkal:

1. Technologické
    a. Nedostatečná výkonnost (např. odezva) EAI nástroje.
    b. Chybný výběr platformy (neposkytuje potřebné nástroje).
    c. Malá škálovatelnost, flexibilita a otevřenost systému.
    d. Chybné technické řešení integračních případů.
    e. Problematická integrace black box aplikací.
2. Organizační
    a. Bobtnání projektu (analýzou integračních případů se objevují další a další potenciální integrační případy).
    b. Nedostatečná spolupráce mezi týmy projektového týmu.
    c. Chybné nebo nedostatečné zadání projektu (nedostatek podkladů pro vyhodnocování řešení integračních případů).
    d. Chybně nastavené vyhodnocení projektu a závislost dodavatele na těchto výsledcích.
    e. Nedostatečná spolupráce zástupců ze strany zákazníka.

Rizika by měla být identifikována již při přípravě business case dokumentu a zároveň k nim být navržena protiopatření. Pokud je není možné přímo eliminovat, pak lze použít techniku předpokladů. Ta spočívá v definici předpokladu, kterým se popisuje předpokládaný stav na základě aktuálních informací, u něhož existuje možnost popsaná rizikem, že se změní. V průběhu projektu se periodicky ověřuje platnost předpokladu ve vztahu k aktuální situaci a jakmile je nalezen nesoulad, pak projektový manažer musí zasáhnout a přizpůsobit projektový plán aktuální situaci.

7.5 Rozhodování

Rozhodnutí o realizaci integračního projektu není jednoduché, jelikož závisí na mnoha faktorech. V zásadě musí rozhodující manažer (decision maker, zpravidla sponzor projektu) zohlednit následující:

- typ integračního projektu (změna celé integrační platformy má samozřejmě větší dopad, než nasazení nové aplikace),
- rozsah projektu (do jaké míry projekt zasáhne podnik),
- porovnání předpokládaných výnosů, nákladů a potenciálních rizik projektu,
- připravenost podniku na procesní přístup EAI,

- současný stav PIS.

Otázkami pro rozhodování o realizaci integračního projektu se strukturovaně zabývá např. [BABU, 2003]. [SCHMIDT, 2003a] představuje souhrn základních oblastí, na které by se měl manažer zaměřit při posuzování připravenosti podniku na EAI projekt:

1. EAI metodika - Jsou v podniku implementované procesy EAI projektu? Jsou dokumentovány, monitorovány, vyhodnocovány?
2. Podnikové procesy - Jsou podnikové procesy dokumentovány a jsou konzistentní? Jsou informační toky automatizované ? Jsou podnikové procesy efektivně řízeny?
3. Systémová architektura - Je definována architektura podnikového systému včetně standardů a frameworku? Je monitorováno využití aplikací a rozhraní ?
4. Koncept integrace - Je implementován koncept integrační platformy ve shodě s EAI?
5. Řízení designu a metadat - Používají se standardizované nástroje a konvence při návrhu systémů a rozhraní? Využívá se úložiště (repository) metadat?

Následující tabulka 4 - Capability Maturity Model ve vztahu k EAI (zdroj: [SCHMIDT, 2002a]) znázorňuje vztahy mezi úrovněmi (vyzrálostí) podnikových procesů, EAI a modelováním v podniku v tzv. modelu CMM (Capability Maturity Model). Modelováním se zde rozumí schopnost využívat modelů a metadat v plánování a rozhodování podniku. Podnik by měl ve svém vývoji udržovat rovnováhu mezi jednotlivými úrovněmi, neboť odchylující se stavy jsou neefektivní (např. nemá smysl investovat do EAI na úrovni 3, pokud se podnikové procesy nacházejí stále na úrovni 2).

Úroveň CMM	Úroveň podnikových procesů	Úroveň EAI	Úroveň modelování
1.	nezávislé	počáteční	statické
2.	sdílené	opakovatelná	aktivní
3.	navazující	holistická	dynamické
4.	řízené	přispívající	předvídající
5.	rozvíjející se	optimalizující se	učící se

tabulka 4 - Capability Maturity Model ve vztahu k EAI (zdroj: [SCHMIDT, 2002a])

8. Požadavky v integračních projektech

Integrační projekty mají specifické požadavky a ty budeme v této kapitole kategorizovat a rámcově charakterizovat.

Požadavek je, dle IEEE standardu 100-1996, "podmínka způsobilosti, která musí být splněna systémem nebo jeho komponentou pro splnění smlouvy, standardu, specifikace nebo dalších formálních dokumentů" (přeloženo z [HORCH, 2003]). Požadavek chápejme jako požadovaný rys, vlastnost nebo chování systému.

Softwarové požadavky se rozlišují dle [LEFFINGWELL, 2005] na:

1. funkční - jak se má systém chovat (pomocí vstupů, výstupů a prováděných funkcí),
2. nefunkční - atributy systému nebo systémového prostředí (např. spolehlivost, dostupnost, výkonnost, podpora),
3. designová omezení - neovlivňují externí chování systému, ale musí být splněny (např. průmyslové standardy, kvalita programového kódu).

Detailně se specifikováním softwarových požadavků zabývá [LEFFINGWELL, 2005].

V této kapitole jsou sepsány skupiny požadavků, které jsou typické pro integrační projekty při nasazování integrační platformy. Zdrojem pro tuto část byly informace z projektů (projektová dokumentace), na kterých jsem se účastnil.

1. **Funkční požadavky** (FREQ - Functional requirements)
   - Seznam procesů, které mají být automatizovány přes integrační platformu (včetně jejich záložní varianty).
   - Požadavky aplikací na integraci.
   - Vymezení dat, která mají být mezi aplikacemi přenášena.
   - Vymezení dat, která mají být uchovávána v integrační platformě (např. v ODS).
   - Metriky pro porovnání stavu integrace aplikací před a po zavedení integrační platformy (např. aplikace X nově zpracuje o 150 procent objednávek více).
   - Preference vybraných zpráv, služeb, procesů.
   - Bezpečnost řešení a soulad s legislativou.
   - Monitorování procesů (např. KPI, stav procesu), možnost zasahovat do běžících procesů.
   - Další požadavky vyplývající z dalších plánovaných projektů nemající přímý vztah k primárnímu zadání projektu (např. dodatečné položky rozhraní).
2. **Výkonnostní požadavky** (PREQ - Performance requirements)
   - Maximální doba předání kritických dat mezi aplikacemi (dvě varianty - automatické předání iniciované událostí ve zdrojové aplikaci, vyžádané - volající aplikace požádá o data volanou aplikaci).
   - Maximální doba trvání integrovaného procesu na straně integrační platformy.
   - U databázových komponent (ODS, DWH) maximální doba odezvy (pro zápis i čtení dat).
   - Garantovaný počet současně běžících procesů při splnění ostatních (především výkonnostních) požadavků (např. 7000 objednávek denně).
   - Garantovaný počet splněných požadavků integrační platformou za jednotku času (např. 2000 žádostí o poskytnutí údajů o zákazníkovi za den s dobou odezvy 0,5 sekundy).
   - Maximální zátěž na integrované aplikace (např. aplikace X, z které integrační platforma čerpá data, bude schopna i po integraci zpracovávat současně 12000 požadavků uživatelů za den (současný stav)).

   Pro výkonnostní požadavky jsou zpravidla na žádost integrátora doplněny předpoklady pro akceptační kritéria, např.:
   - Nulová nebo definovaná doba odezvy integrovaných aplikací při volání integrační platformou (aby bylo možno měřit čas spotřebovaný integrační platformou).

- Průměrné a maximální množství protékajících dat (např. zpráv).
- Předpokládané množství ukládaných dat (v ODS, DWH).

3. **Požadavky na monitorování a administraci** (MREQ - Monitoring requirements)
   - Seznam událostí, které mají být sledovány (včetně bezpečnostních incidentů) a způsob jejich sledování.
   - Způsob provozní administrace systému (např. řešení chybových stavů procesů).
   - Způsob notifikace administrátorů při krizových událostech.
   - Detekce úzkých míst systému.
   - Přístupová práva k systému.
4. **Požadavky na dostupnost systému** (AREQ - Availability requirements)
   - Akceptovatelný způsob obnovení systému.
   - Akceptovatelný způsob náběhu systému (podklady pro migrační plán).
   - Garantovaná dostupnost celého řešení (např. 99,9%).
   - Schopnost automatického zotavení systému.
   - Schopnost upgradování systému za provozu s definovaným dopadem na poskytované služby.
5. **Ostatní požadavky** (OREQ - Others requirements)
   - Standardy a best practices pro ODS, DWH a další databázové prvky (viz např. [TMF, 2003] a [REILLY, 2004]).
   - Standardy a best practices pro modelování integrační procesů (kanonický formát zpráv, standardy webových služeb, vzory pro zpracování zpráv, atd.).
   - Definice vývojových standardů.
   - Vlastnictví zdrojových kódů a možnost jejich úpravy.
   - Požadavky na postimplementační podporu řešení.
   - Dokumentace.

Tyto požadavky jsou zpravidla uvedeny v příloze ke smlouvě a ke každému z nich jsou uvedena přesná akceptační kritéria (ta říkají, za jakých podmínek je požadavek zákazníkem akceptován). Přesná a včasná specifikace požadavků výrazně snižuje neproduktivní náklady a nastavuje jasný vztah mezi zákazníkem a integrátorem. Požadavky na straně zákazníka musí definovat tým s hlubokou znalostí potřeb podniku a odpovídající znalostí možností jejich řešení (k těmto znalostem je možno najmout konzultační firmu).

Je obvyklé, že požadavky zákazníka není možné současně splnit všechny nebo jen za určitých předpokladů (ty jsou definovány rovněž jako dodatek smlouvy). Proto se požadavky postupně upřesňují podle stupně poznání možností v projektu.

V [LOSAVIO, 2003] je popsán model standardní kvality pro EAI middleware. Vychází z modelu kvality ISO 9126-1 a kombinuje jej s modelem EAI middlewaru z [CC02] - viz obrázek 1 - Rozšířený EAI framework (zdroj: [LOSAVIO, 2005]). Definuje 5 hlavních kvalitativních charakteristik EAI:

1. Funkcionalita (=> FREQ)
   a. interoperabilita
   b. bezpečnost
   c. soulad se standardy a legislativou
2. Spolehlivost (=> AREQ)
   a. odolnost vůči chybám
   b. vyzrálost
   c. obnovitelnost
   d. dostupnost
   e. soulad se standardy a legislativou
3. Výkonnost (=> PREQ)
   a. časové charakteristiky (výkonnost)
   b. využití zdrojů

    c. soulad se standardy a legislativou
4. Udržovatelnost (=> MREQ)
    a. upravitelnost
    b. analyzovatelnost
    c. soulad se standardy a legislativou
5. Přenositelnost (=> OREQ)
    a. adaptabilita
    b. koexistence s ostatními systémy
    c. soulad se standardy a legislativou

Porovnáme-li tyto charakteristiky s výše popsanými požadavky, tak vidíme, že oblasti kvalitativních charakteristik odpovídají skupinám požadavků (odkaz na požadavky je uveden v závorce u charakteristik), což zpětně verifikuje kompletnost popsaných požadavků.

9. Integrační proces a integrační případ

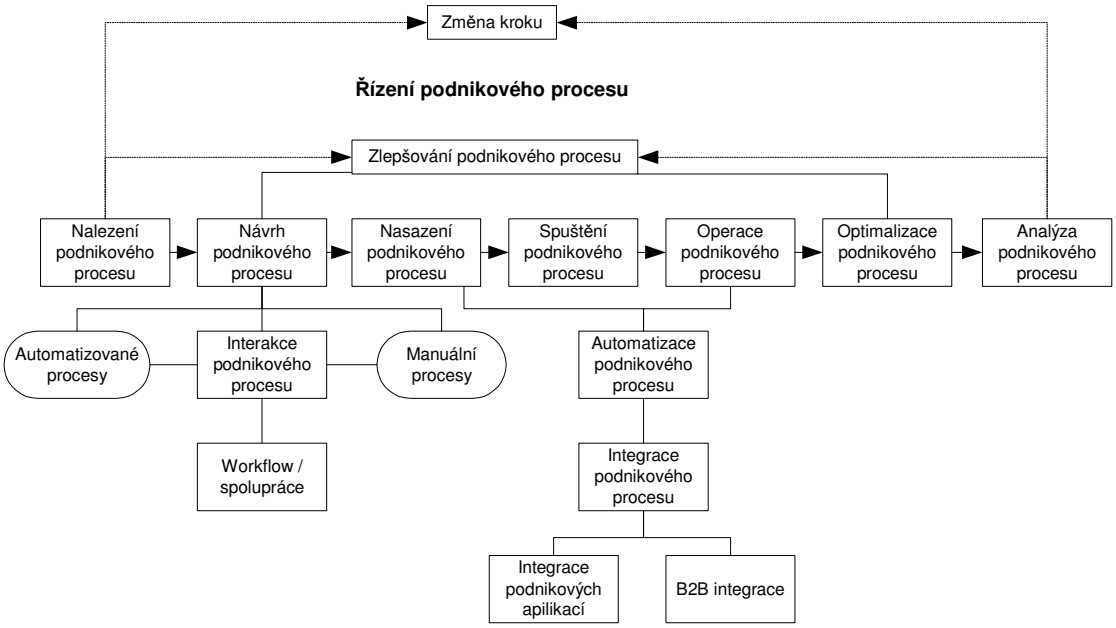
Tato kapitola představuje přístup k analýze integračních úloh založený na integračních procesech a případech. Popisuje způsob, jakou jsou změny z podnikových procesů transformovány do integračních procesů ve fázi analýzy integračního projektu, a jak jsou ve fázi návrhu integrační procesy překlopeny do integračních postupů, jako most mezi těmito fázemi používá specifikaci integračního případu. Zároveň upozorňuje na změnu orientace z procesního do technologického světa, ke které dochází při přechodu z analýzy do návrhu.

9.1 Vztah BPR a EAI

Projekty BPR (Business Process Reengineering) analyzují stávající podnikové procesy a navrhují jejich změny. Na realizaci podnikových procesů se do značné míry může podílet (a je všeobecný trend tento podíl maximalizovat) podnikový informační systém (ve smyslu systému zahrnujícího integrované podnikové aplikace). V souvislosti s tím se většina metodik BPR se ve fázi návrhu procesů zabývá i jejich implementací pomocí IT ([ŘEPA, 2006a] uvádí takto orientované metodiky T. Davenporta, Manganelliho a Kleina, ARIS (Architecture of Integrated Information Systems) prof. Scheera).

Generuje-li tedy BPR projekt změny do podnikových procesů, znamená to tedy nevyhnutelně i zásahy do informačního systému podniku. [ŘEPA, 2006a] klasifikuje procesní změny na vnitrofunkční, mezifunkční a meziorganizační. Za zjednodušujícího předpokladu, že ucelená funkční oblast v podniku je pokryta právě jednou podnikovou aplikací, pak můžeme toto rozdělení transformovat takto:

- vnitrofunkční změna procesu -> změna v rámci jedné aplikace,
- mezifunkční změna procesu -> změna ve vnitropodnikové integraci aplikací,
- meziorganizační změna procesu -> vněpodniková integrace aplikací (B2B).



obrázek 15 - Modelování podnikových procesů (zdroj: [STRÜVER, 2002])

Podle [ŘEPA, 2006a] je předmětem obecné fáze designu projektu BPR mimo další aktivity související s BPR i detailní definice procesů a vývoj informačního systému. To je právě ta fáze, v které mohou vznikat podřízené projekty s cílem integrace podnikových aplikací (vedle projektů pro implementaci procesů do podnikových aplikací a B2B projektů) -

viz obrázek 15 - Modelování podnikových procesů (zdroj: [STRÜVER, 2002]), kterými se tato práce zabývá. Dalšími dvěma typy projektů se zde nebudeme zabývat.

S vazbou BPR a EAI pracuje i metodika EIM a metodika pro vývoj integrovaných IT infrastruktur (viz kapitola 1.6 Aktuální stav ve světě). V [OBA, 2001] je popsán vztah BPR a EAI na modelovém workflow systému pro podnikové procesy postaveném na technologii CORBA. Oproti této práci popisuje tento vztah ze strany BPR a pouze na úzce profilové technologii.

9.2 Integrační proces

Integračním procesem definujme posloupnost operací, které zajišťují komunikaci s integrovanými aplikacemi, jsou řízeny integrační platformou a mají jeden společný cíl, splnění integračního požadavku. Integrační proces má jasně definovaný začátek (zpravidla příchozí zpráva nebo volání z integrované aplikace, méně často spuštění procesu dle časového plánu) a konec.

V užším pojetí se integračním procesem chápe proces realizovaný na integračním BPI serveru (Business Process Integration server) komponentou BPM (Business Process Management), v širším pojetí si pod ním představíme integraci prostřednictvím kterékoliv integrační komponenty (EIP (Enterprise Information Portal), BPI, EII (Enterprise Information Integration), ODS (Operational Data Store), DWH (Data Warehouse) i přímou komunikací).

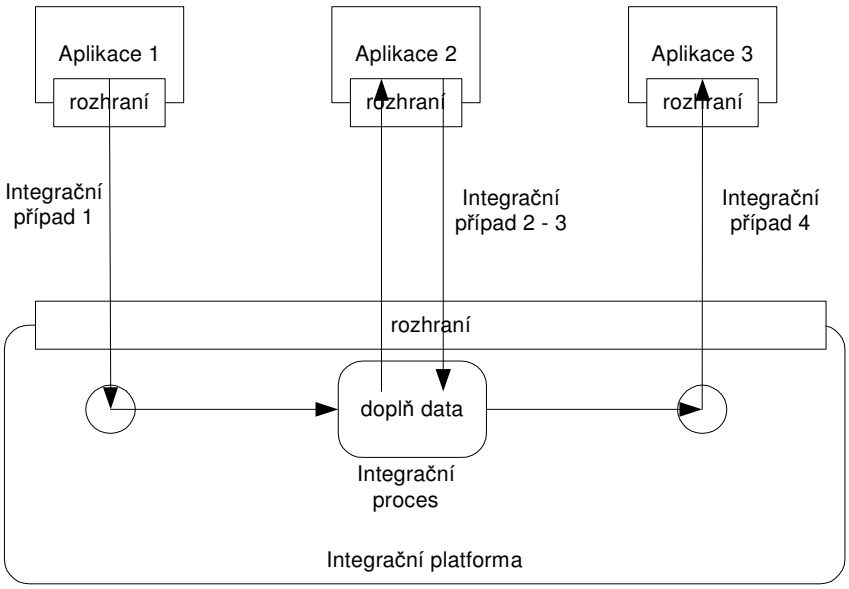
Integračním případem označujme elementární komunikační operaci mezi dvěma integrovanými aplikacemi, nebo aplikací a integrační komponentou. Integrační proces se může skládat z více integračních případů.

Představme si nyní jednoduchý podnikový proces přijetí objednávky. Tento proces bude podporován v informačním systému třemi aplikacemi. První bude objednávky přijímat, druhá bude udržovat informace o zákaznících, které budou do objednávek pro potřeby podniku doplňovány, a třetí, která bude objednávky evidovat a dále je zpracovávat (zde by mohl proces dále pokračovat, ale to není pro naše účely podstatné). Dosud byl tento proces podporován izolovaně jednotlivými aplikacemi a vzájemná konsolidace a předávání informací mezi aplikacemi probíhalo manuálně. V důsledku změny podpory procesu informačními technologiemi se řeší integrační úloha automatizovaného online předávání dat mezi aplikacemi.

Obrázek 16 - Vztah mezi integračním procesem a případem (zdroj: autor) popisuje plánovaný stav ve formě integračního procesu. Aplikace 1 proces iniciuje např. zasláním zprávy (založení objednávky) do integrační platformy (integrační případ 1), integrační proces si vyžádá doplnění dat do objednávky od aplikace 2 (integrační případ 2 - 3) a následně požádá aplikaci 3 o vyhotovení objednávky. Proces by se samozřejmě mohl větvit (např. zrušení objednávky), nebo pokračovat (monitorování stavu objednávky). Integrační případ 2-3 znamená rozdíl mezi synchronní a asynchronní komunikací - u synchronní komunikace se bude jednat pouze o jeden integrační případ u asynchronní o případy dva.

Povšimněme si, že vzorový příklad neříká nic o technologie integrace. Mohlo by se jednat o realizaci prostřednictvím BPM komponenty integračního serveru stejně jako prostřednictvím ODS a jeho rozhraní. Tím se zabývá až další fáze, fáze návrhu.

Integrační proces může být zcela elementární - zavolání jedné aplikace přesměruje integrační platforma na volání druhé aplikace a to je celý proces. Vedle toho mohou vznikat komplexní integrační procesy obsahující složité rozhodovací stromy a návraty do předchozích stavů, sestávající s desítek integračních případů. Na druhé straně není proces nic jiného než složitá funkce (či procedura, služba), která se skládá z jednotlivých kroků a jejíž běh může být snadno sledován.



obrázek 16 - Vztah mezi integračním procesem a případem (zdroj: autor)

Počátky integračního procesu, resp. případu hledejme v analýze integračních požadavků, kde se identifikují možné oblasti integrace aplikací. Tato identifikace se provádí na základě procesního modelu, z kterého vyčteme, které procesy na sebe jak navazují, a jakým způsobem jsou podporovány aplikacemi. Opomineme-li integrační projekty z čistě technologických důvodů (např. výměna integrační platformy), pak jejich zadání by vždy mělo vycházet ze změn podnikových procesů.

9.3 Analýza integračního případu

Integrační případ je elementární případ integrace mezi aplikacemi a je klíčem pro pochopení souvislostí a složitosti integrace, proto se mu budeme věnovat detailně. Ta část rozhraní aplikace, která se využívá pro konkrétní integrační případ (např. procedura) se abstraktně nazývá integračním bodem.

V detailní **analýze aplikací** se zkoumá zejména jejich datový a funkční model. Ve [VOŘÍŠEK, 1997] je přesně definován koncept vrstvené architektury, která je jedním z klíčů pro integraci. Je pochopitelně mnohem snazší integrovat aplikaci, která má jasně definované vrstvy funkční architektury, což v době návrhu integračních vazeb podstatně zjednoduší a zrychlí vytváření adaptérů na aplikaci. Specifickou kapitolu tvoří aplikace, ke kterým neexistuje funkční ani datový model, resp. dokumentace, případně není dostupné žádné API (Application Programming Interface), pomocí kterého by bylo možno aplikaci připravit na integraci (tyto aplikace označujeme black-box aplikacemi).

Rozhraní aplikací můžeme rozdělit na tyto typy (viz také kapitola 10.4.1 Integrační rozhraní):

1. Datové - aplikace poskytuje přímý přístup k vybraným datovým entitám (např. pro provádění snapshotů).
2. Funkční - aplikace poskytuje přístup k funkcím (službám). Integrace se provádí na úrovni funkcí, nikoliv dat (integrace tedy neprobíhá tak, aby se definovala část dat, která budou přístupná jiným aplikacím, ale vždy se definují funkce, které nad těmito daty budou pracovat). V době návrhu je třeba definovat, které služby bude třeba do aplikace doprogramovat. V zásadě lze rozlišit služby na ty, které jsou v aplikaci již implementovány a nejsou pouze nabízeny jiným aplikacím (typicky třeba založení objednávky), a na ty, které dosud implementovány nebyly, neboť nebyly potřeba.

S příchodem možnosti integrace se zpravidla objeví nové požadavky na další funkce, které by měla aplikace zajišťovat pro jiné aplikace. Typicky se může jednat např. o vytváření analytických dotazů nad daty, které patří aplikaci, avšak byly by vhodným vstupem pro jinou aplikaci, která se zabývá zpracováním údajů pro marketing.
Z pohledu integračního architekta je nutné zachovat logickou souvislost dat a nad nimi realizovaných funkcí, pokud je to možné. Není tedy vhodné aplikovat myšlenku typu, pokud tato analytické funkce nad daty jiné aplikace slouží pouze pro marketingovou aplikaci, budeme tuto funkci implementovat do marketingové aplikace. Tento přístup je nutný z důvodu zachování jednoho "vlastníka" dat (objektový přístup), tedy aplikace, který by měl mít vždy možnost sledovat či dohledat, jaké operace jsou nebo byly nad daty prováděny. Dalším dobrým důvodem pro zachování tohoto principu je příprava na možnost, že danou funkci v budoucnu budou využívat i další aplikace, které např. nahradí naši marketingovou aplikaci a my bychom museli službu znovu definovat a programovat.
Všechny takto definované služby (řádově se u velkého podniku jedná o desítky až stovky služeb) včetně parametrů je nutno uspořádat do logických celků a definovat pro ně rozhraní, které bude později vstupem pro tvorbu adaptéru.

3. Prezentační - aplikace poskytuje přístup ke své prezentační vrstvě.

Rozhodnutí o typu rozhraní použitého pro daný integrační případ je premisou pro analýzu a popis integračního případu na detailní úrovni. Toto rozhodnutí je záležitostí architekta, a zde se tedy prolíná analýza a návrh řešení.

U větších projektů se doporučuje udržovat základní seznam integračních případů, který by měl obsahovat tyto přehledové informace:

- Identifikace integračního případů.
- Identifikace rozhraní mezi aplikacemi.
- Směr přenosu dat (rozumí se hlavních dat, nikoli kontrolních, doplňujících apod.).
- Iniciující (volající) aplikace.
- Iniciující událost.
- Stručný popis integračního případu.
- Způsob realizace (např. snapshot, textový soubor, zpráva, služba).
- Požadován souběh se současným řešením?
- Fáze projektu, kdy bude implementován.
- Vazba na další integrační případy.

Detailní popis integračního případu se rozpadá na tři části:

- API zdrojové aplikace,
- API cílové aplikace,
- popis zpracování integrační platformou.

[HAUBRICH, 2005] se rovněž zabývá detailní analýzou integračního případu v podobě integračního problému, který strukturuje na zdrojovou komponentu a cílový systém a na nižší úrovni na procesní architekturu, funkční a sémantický soulad a aplikační architekturu. V níže uvedeném členění odpovídá aplikační architektura obsahově popisu API aplikací a ostatní části jsou popsány ve zpracování integrační platformou. Níže uvedený popis je proveden na výrazně detailnější úrovni než v [HAUBRICH, 2005].

Tento popis je už částečně závislý na zvoleném integračním postupu, neboť každý z nich potřebuje trochu odlišné informace. U datové integrace je to typicky popis integrovaných datových struktur, charakteristika dávkových přenosů a dávkového zpracování na integrační platformě, u funkční integrace je to detailní popis služeb aplikací a chybových stavů a online zpracování na integrační platformě, u integrace na prezentační úrovni se nad rámec funkční integrace navíc zkoumá i formátování prezentační vrstvy (viz také [MAVERICK, 2003]). Uvádím přehled základních atributů detailního popisu:

1. Integrační případ datový:
    a. API aplikací (volající, volaná):
        - identifikace databázového uživatele a schéma,
        - název a popis funkcionality databázové procedury, nebo jobu,
        - popis chybových kódů,
        - popis datových položek (včetně formátu a struktury dat):
            - název databázové tabulky,
            - název datové položky,
            - popis datové položky,
            - typ datové položky,
            - označení, zda vstupní nebo výstupní datová položka,
            - označení, zda je datová položka povinná,
        - doplňující informace (znaková sada, omezující podmínky datových položek apod.).
    b. Integrační platforma (zpravidla ODS nebo point-to-point přístup):
        - informace o integračním případu,
            - identifikace integračního případu,
            - identifikace rozhraní,
            - související integrační případy,
            - funkční popis integračního případu s komunikačním schématem,
            - iniciující aplikace,
            - iniciační událost,
            - zdrojová aplikace (ve smyslu poskytující hlavní data),
            - cílová aplikace,
            - požadovaná doba odezvy,
            - předpokládaný objem dat za přenos,
            - předpoklady (např. naplněné vazební tabulky, namigrovaná data, atd.),
        - detailní popis komunikace mezi aplikacemi (volání procedur, vyhodnocování atd.),
        - mapování datových položek mezi aplikacemi a požadované transformace,
        - doplňující informace (např. maximální obsah jedné dávky dat),
        - řešení chybových stavů.
2. Integrační případ funkční
    a. API aplikací (volající, volaná):
        - identifikace aplikace,
        - název a způsob volání a popis funkcionality služby aplikace,
        - popis chybových kódů,
        - popis předávaných parametrů (včetně formátu a struktury dat):
            - název parametru,
            - popis parametru,
            - typ parametru,
            - označení, zda vstupní nebo výstupní parametr,
            - označení, zda je parametru povinná,
        - popis společných speciálních hodnot parametrů (např. zástupné hodnoty),
        - doplňující informace (znaková sada, omezující podmínky parametrů apod.).
    b. Integrační platforma (zpravidla message broker nebo integrační server)
        - informace o integračním případu:
            - identifikace integračního případu,

        - identifikace rozhraní,
        - související integrační případy,
        - funkční popis integračního případu s komunikačním schématem,
        - iniciující aplikace,
        - iniciační událost,
        - zdrojová aplikace (ve smyslu poskytující hlavní data),
        - cílová aplikace,
        - požadovaná doba odezvy,
        - předpokládaný počet přenosů za den,
        - předpoklady,
    - workflow na integrační platformě (grafické schéma za použití např. BPMN (Business Process Modeling Notation, [OMG, 2006])):
        - iniciační událost API volající aplikace,
        - přijetí požadavku,
        - transformace požadavku, zpracování parametrů určených pro integrační platformu,
        - volání API volané aplikace,
        - přijetí odpovědi,
        - transformace odpovědi, zpracování parametrů určených pro integrační platformu,
        - odeslání odpovědi na API volající aplikace,
    - mapování datových položek mezi aplikacemi a požadované transformace (odkazy na workflow),
    - doplňující informace (např. maximální doba trvání workflow),
    - řešení chybových stavů.
3. Integrační případ prezentační - je prakticky shodný s funkčním, jen se navíc definuje formátování prezentační vrstvy.

Informační zdroje [IBM] a [HOHPE, 2003a]) obsahují základní klasifikaci integračních případů podle řešené integrační úlohy:

1. Integrační případy datové:
    a. Propagace - integrační platforma distribuuje data ze zdrojové aplikace do předem definovaných cílových aplikací.
    b. Replikace - integrační platforma zajišťuje konzistenci dat mezi oběma aplikacemi.
    c. ODS - integrační platforma slouží jako agregátor a skladiště dat ze zdrojových aplikací a zdroj agregovaných dat pro cílové aplikace.
    d. Společná repository (Federated repository) - integrační platforma slouží jako on-line dotazovací rozhraní pro volající aplikace, který dotaz rozdělí dle pravidel na jednotlivé dotazy pro volané aplikace a vrátí konsolidovanou odpověď (viz integrační přístup virtuální databáze).
2. Integrační případy funkční:
    a. Agregátor (Aggregator) - integrační platforma slouží jako konsolidátor dat z více aplikací do jedné zprávy určené jako odpověď na žádost volající aplikace.
    b. Směrovač (Router) - integrační platforma na základě obsahu (včetně metadat) zprávy předá zprávu příslušné volané aplikace (tedy platforma sama rozhoduje dle pravidel o tom, které aplikaci zprávu předá) a odpověď vrátí volající aplikaci.
    c. Transakce (Transactional) - integrační platforma zajistí provedení celé transakční operace (napříč aplikacemi), v případě neúspěchu provede návrat (rollback) do původního stavu.

d. Zprostředkovatel (Broker) - integrační platforma zajistí provedení operací na žádost volající aplikace ve volaných aplikacích (pokud např. nejsou dostupné, pak zařadí požadavek do fronty pro příslušnou aplikaci).
e. Řízený proces (Managed Process) - integrační platforma na základě události z volající aplikace provede dle definovaných business pravidel volání příslušných služeb aplikací.

V průzkumu [MACVITTIE, 2003] se uvádí, že nejčastěji se řeší integrace mezi legacy aplikacemi.

9.4 Výběr integračního postupu

Ve fázi návrhu se jednotlivým integračním případům přiřazují integrační postupy, pomocí kterých budou implementovány. Tento proces je pro architekta poměrně náročný, neboť na jedné straně má integrační případ s řadou požadovaných charakteristik a na straně druhé rozsáhlou sadu integračních postupů.

V prvé řadě se musí řídit podnikovou integrační strategií (viz kapitola 10.3 Integrační koncepce). V rámci integrační strategie jsou definovány klíčové integrační komponenty, které společně tvoří integrační platformu podniku - typicky se může jednat např. o kombinaci BPI, EIP, ODS a DWH. Spoluprací těchto komponent a podnikových aplikací vznikají tzv. integrační cesty, po kterých protékají data mezi aplikacemi. Podle počtu integračních komponent, které jsou zaimplementovány do integrační platformy podniku, se rozšiřuje portfolio integračních cest, z nichž architekt může vybírat vhodný způsob řešení konkrétního integračního případu.

Uvažujeme-li například, že se integrační platforma skládá z ODS a BPI, pak integraci dvou aplikací může obecně řešit 4 způsoby - datově přes ODS nebo přímo (např. datovou replikací) mezi aplikacemi a nebo prostřednictvím funkčních rozhraní přes BPI nebo opět přímo (např. pomocí RPC).

Vedle toho má architekt k dispozici pomyslný několika úrovňový rozhodovací strom integračních postupů (viz kapitola 10.7 Rozhodovací strom integračních postupů), podle kterého lze jednoznačně určit nejvhodnější integrační přístup a (podle dalších rozhodovacích úrovní) technologii pro daný integrační případ. Architekt z rozhodovacího stromu vyřadí ty integrační postupy, které nejsou integrační platformou podporovány (neexistují integrační cesty), a následně s pomocí stromu stanoví nejvhodnější integrační technologii pro integrační případy.

9.5 Modelování procesů

Výběrem technologie proces návrhu teprve začíná, neboť je třeba tvůrčím přístupem integrační případ přenést do technologie, tedy navrhnout datové položky a způsob jejich zpracování v aplikacích a integračních komponentách.

Vedle funkcionality poskytované rozhraními aplikací leží jádro integrace na integrační komponentě. Integrační platforma přijme požadavek (ve formě události nebo volání), zpracuje jej dle obchodní logiky a transformovaný jej předá další aplikaci.

Obchodní logika může být zakódovaná přímo ve zdrojovém kódu vyvinutým pro integrační platformu. Integrační komponenty typu integračních serverů nabízejí vlastní vizuální vývojové prostředí (např. BEA Weblogic Workshop), kde lze proces zpracování částečně modelovat a částečně programovat. Tyto nástroje umí automaticky detekovat rozhraní integrovaných aplikací a nabízejí jeho funkcionalitu (např. služby) pro přímé zakomponování do procesu (v terminologii Microsoftu se tato činnost nazývá orchestrace). Orchestraci webových služeb dle [NEWCOMER, 2004] podporují dva konkurenční standardy - Business Process Execution Language (WS-BPEL) a Choreography Description Language (WS-CDL). Stejný zdroj se zabývá orchestrací na detailní úrovni. Modelováním procesů se

zabývá např. [VONDRÁK, 2004], [VOŘÍŠEK], [BUSSLER, 2003] a [ŘEPA, 2006b]. [ŘEPA, 2006b] se rovněž zabývá vztahem a závislostmi mezi procesním modelováním a modelování v UML.

[BUSSLER, 2003] definuje **elementy pro procesní modelování** tyto:

1. Událost - oznámení o situaci, která je relevantní pro integrační proces. [BUSSLER, 2003] rozpracovává událost typu příchozí zprávy, ale stejně tak se může jednat např. o uplynutí určité doby. Událost má vlastní životní cyklus, jehož stavy jsou ovlivňovány především integračním procesem.
2. Business objekt - abstrakce, která reprezentuje ucelenou sadu dat a služeb ([BUSSLER, 2003] uvádí jen data), které se vztahují k logickému prvku v procesu - např. objednávce.
3. (Integrační) proces - posloupnost operací, které reagují na vstupní událost. Proces má rovněž vlastní životní cyklus - posloupnost stavů. [BUSSLER, 2003] rozlišuje tři typy podprocesů:
    a. Proces rozhraní - zajišťuje zpracování vstupů a výstupů (např. jejich korelaci) na rozhraní.
    b. Business proces (zde se označuje pouze jako součást integračního procesu) - obsahuje obchodní logiku (rozhodování dle business pravidel) procesu.
    c. Vazební proces - provazuje business procesy a procesy rozhraní.
4. Integrační bod (Endpoint) - element integračního rozhraní, skrze který aplikace komunikuje se svým okolím, např. konkrétní služba rozhraní.
5. Řešení chyb (error handling) - ošetřuje neočekávané stavy procesu.

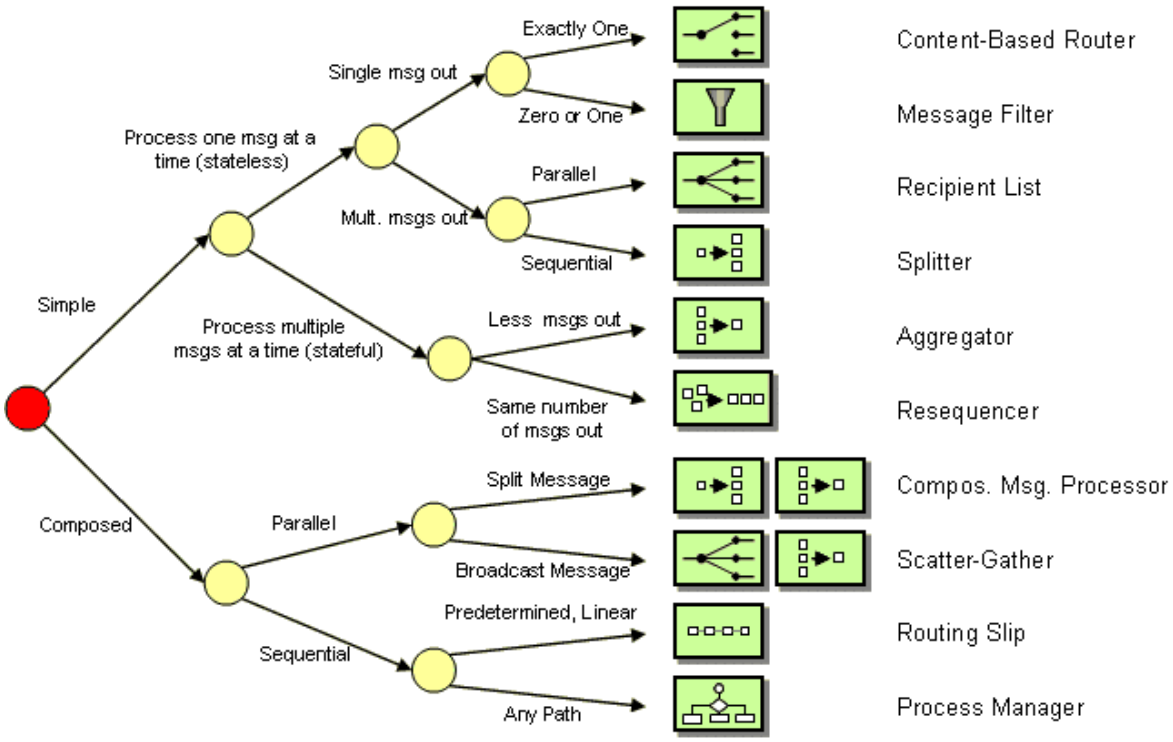
Detailně se modelováním procesů zabývá také [JOHANNESSON, 2001], který pro jejich návrh používá BML (Business Model Language), který poskytuje notaci pro události, hlavní a kontrolní činnosti a člověka a aplikace jako externí prvky. [JOHANNESSON, 2001] rovněž definuje různé typy zpráv (žádost o (a potvrzení) informace, služby, rezervace a objednání, příslib služby, notifikace, zrušení rezervace a objednání) a procesů (zákaznický, rozhraní, žádací, uvolňovací, synchronizační, údržbový a aktualizační). Pro modelování integračních procesů doporučuji používat příbuzné **BPMN** a **BPML** (Business Process Modeling Language), neboť jsou obecně uznávanými standardy.

Pro zpracování integračních případů v BPI ve formě zasílání zpráv doporučuji [HOHPE, 2003b], či [HOHPE], kde jsou definovány univerzální vzory návrhu (design patterns). Obecnější vzory definuje např. [IBM].

[HOHPE] se velmi přehledně a detailně zabývá výběrem typických způsobů zpracování na integrační platformě (primárně pro zasílání zpráv) typu integrační server a na obrázku 17 - Vzory zpracování integrační případů na integračním serveru (zdroj: [HOHPE]) sumarizuje typické vzory zpracování integrační případů.

[OMG, 2001c] specifikuje využití **UML** pro modelování integrace aplikací (EAI), což umožňuje použít UML modelovací nástroje ve fázi návrhu. Omezuje se však pouze na událostmi řízenou integraci. Definuje dva hlavní metamodely, prvním z nich je EAI Integration Metamodel, který je specializací Flow Composition Modelu (FCM), který je popsán v [OMG, 2001a]. Přebírá z něj koncepci toků, tokových uzlů a kompozice a přidává základní koncepce asynchronní komunikace, frontování zpráv a obsahu a formátu zpráv. Druhým metamodelem je EAI Common Application Metamodel (CAM), který je určen pro popis rozhraní integrovaných systémů. Dále pak definuje tzv. UML profil pro EAI, což je sada notací v UML, které popisují modelové prvky specifické pro EAI. Modelování integračních procesů je popsáno v samostatné kapitole s použitím modelů aktivit. Použití profilu je demonstrováno mapováním na produkt IBM Websphere MQ (dříve MQSeries) a WebSphere MQ Integrator. Příbuzným zdrojem je zmíněný [OMG, 2001a], který obsahuje UML profil pro EDOC (Enterprise Distributed Object Computing). Jeho obsahem je modelování spolupráce

distribuovaných komponent a podnikových procesů, popis ECA (Enterprise Collaboration Architecture) (viz také [OMG, 2004a]) založené právě na distribuovaných objektech a mapování do technologií webových služeb, EJB a CORBA.



obrázek 17 - Vzory zpracování integrační případů na integračním serveru (zdroj: [HOHPE])

Obchodní logika může tedy být vizuálně znázorněna nástrojem pro modelování procesů, navíc s využitím typických vzorů. I takto by však zůstal ve velké části (neboť namodelované procesy nejsou mezi konkurenčními vývojovými nástroji příliš přenosné (viz také [MATĚJŮ, 2005]), přestože existují domluvené standardy pro jejich exportování a importování - BPML / BPEL (Business Process Execution Language), WSCI - Web Services Choreography Interface) způsob zpracování skrytý ve vygenerovaném kódu. To stěžuje jeho udržovatelnost, proto je snaha mít obchodní logiku definovanou centrálně, aby se změna na jednom místě okamžitě a automaticky aplikovala na všechna potřebná místa, a přehledně, aby bylo možné velmi rychle dohledat, jak je obchodní logika nastavena. Proto se rozvíjí další řada nástrojů, tzv. BRE (Business Rules Engine), které umožňují externě definovat a provozovat tzv. business pravidla (pravidla rozhodování specifická pro podnik, implementující interní know-how) (viz kapitola 13.1 Business rules engine).

Dalším zásadním doporučeným vzorem při zasílání zpráv je **kanonický datový model**. Jeho podstatou je udržovat společný, aplikačně nezávislý formát zprávy pro jednotlivé datové entity. Posílá-li aplikace tedy data o zákazníkovi (je jedno, zda pouze jeden jeho atribut, nebo všechny), vždy je zašle ve struktuře, kterou používají všechny aplikace. O kanonickém datovém modelu dále pojednává [HOHPE, 2003b] a zmiňuje se o něm i [KAYE, 2003] a [CHAPPELL, 2004]. Pro převod zpráv do kanonického tvaru je možné využít rozhraní aplikace, nebo externí komponentu provádějící překlad pro všechny aplikace, nebo kanonický tvar implementovat vnitřně do aplikací.

[JUŘEK, 2004] k použití kanonického tvaru říká : "... se používá technika takzvaného kanonického schématu, což je jediné referenční schéma pro danou entitu. Mezi jednotlivými schématy se pak dokumenty převádí za použití kanonického schématu jako mezistupně.

Pokud oprášíte matematiku ze základní školy, mělo by vám vyjít, že při N různých schématech a jednom kanonickém je počet všech potřebných jednosměrných transformací mezi nimi roven 2*N. Srovnáním tedy snadno zjistíme, že již při 4 různých schématech se nám použití kanonického schématu vyplatí."

Kanonický formát zpráv lze přeneseně aplikovat i na přenos datových entit na úrovni datových rozhraní. S unifikací na úrovni datových položek se setkáváme v ODS ([TMF, 2003] definuje tzv. **SID** (Shared Information/Data Model) model), přechod mezi ODS a kanonickými zprávami popisuje formou rozhraní **OSSJ** ([REILLY, 2004]). Unifikovaný objektový model popisuje [ARLOW, 2003], vzory pro přístup k datům v databázi pro interakci v objektově orientovaných aplikacích definuje [NOCK, 2003].

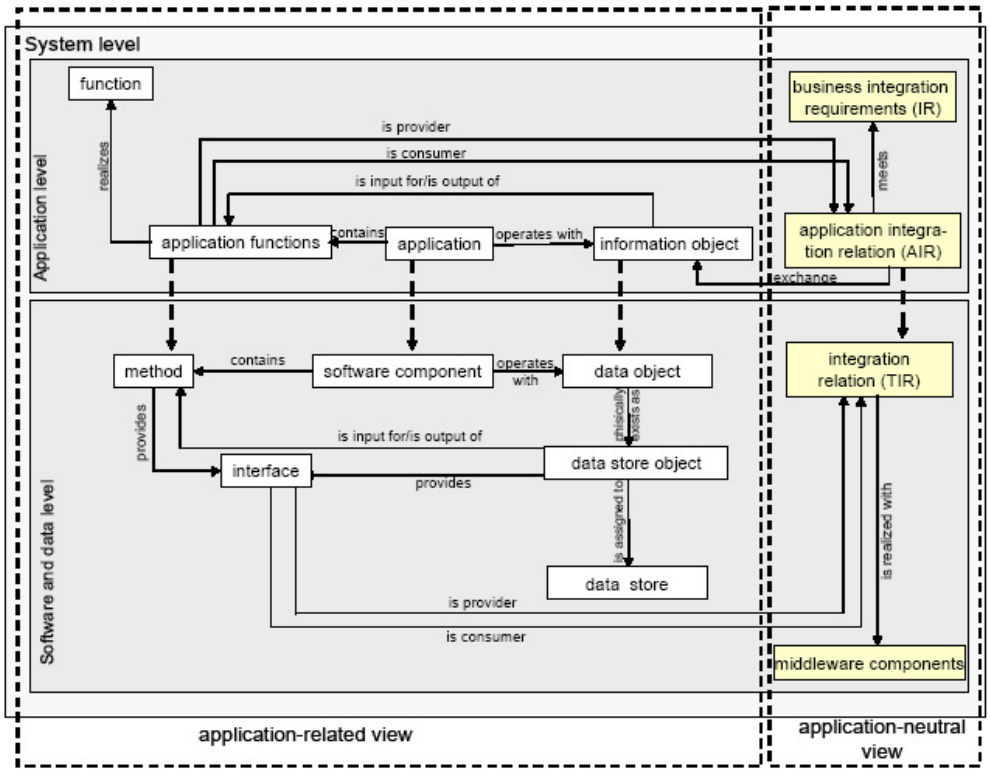
V souvislosti s používáním vzorů při detailním návrhu nelze opomenout **MDA** (Model Driven Architecture), což je objektově orientovaný přístup vypracovaný OMG (Object Management Group). Při návrhu se postupuje modelováním od CIM (Computionally Independent Model) přes PIM (Platform Independent Model) a PSM (Platform Specific Model) až k implementačnímu modelu. Hlavním principem MDA je využití archetypových vzorů při přechodu mezi modely, což obecně výrazně zkracuje dobu návrhu a vývoje. O MDA v souvislosti s integrací aplikací pojednává také [BUCHALCEOVÁ, 2003]. [ARLOW, 2003] uvádí vzory pro objekty typu Party, PartyRelationship, Customer, Product, Inventory, Order, Quality, Money a Rule. [BUCHALCEOVÁ, 2005] shledává metodiku MMDIS a MDA vzájemně kompatibilní, neboť obě vycházejí z postupného modelování od logického k fyzickému modelu a navzájem se neomezují. Totéž platí i pro metodiku integračních projektů.

Vedle návrhu transformačních procesů na integrační platformě se souběžně **navrhují i rozhraní integrovaných aplikací**. Možností jejich návrhu se zde zabývat nebudeme, neboť svým rozsahem překračují rámec této práce (viz např. [YACOUB, 2003], [SHALLOWAY, 2004]). [PYROVOLAKIS, 2003] k tomu poznamenává, že "...implementace aplikací orientovaných na služby předpokládá, že podniky změní metody, nástroje a role vývoje aplikací. Při přechodu z tradičních metodik (např. Yourdon, SSADM) k objektově orientovaným s podporou UML a automatizaci procesů budou muset vyvinout koexistenční strategie."

Pro analýzu a návrh rozhraní integrovaných aplikací vytváří [SCHELP, 2005] model (viz obrázek 18 - Rozšířený referenční model podnikového inženýrství pro integrační účely (zdroj: [SCHELP, 2005])), který popisuje několik vztahů:

1. vztah mezi logickou (application level) a fyzickou (software and data level) úrovní aplikace,
2. vztah mezi funkční (aplikační funkce a metody) a datovou (informační objekty a datové objekty) vrstvou aplikace (prezentační vrstva aplikací není zmíněna),
3. vztah mezi integračními požadavky, integračními vztahy mezi aplikacemi na logické úrovni (AIR - Application Integration Relationship) a na fyzické úrovni (TIR - Technical Integration Relationship) a integračními komponentami (middleware components) a
4. vztah mezi aplikací a integračními vztahy (AIR a TIR).

Koncepčně tento model tvoří obecnou vazbu mezi podnikovými aplikacemi a integračními komponentami a je určitou **protiváhou k modelu integračních cest** (viz kapitola 11.2 Komplexní integrační model PIS), který se naopak zaměřuje primárně na integrační komponenty.



obrázek 18 - Rozšířený referenční model podnikového inženýrství pro integrační účely (zdroj: [SCHELP, 2005])

9.6 Sdílené prvky

Zatímco ve fázi analýzy jsme integrační proces rozdělovali na jeho elementární prvky, ve fázi návrhu je naopak snaha je sdružovat do sdílených celků, které v důsledku snižují náklady na vývoj, testování a následnou údržbu. To se týká sdružování funkcí (či dat) poskytovaných (přijímaných) aplikací do logických celků z hlediska implementace, čili je možné definovat univerzální API aplikace. Rovněž tak způsob zpracování integrační platformou (včetně adaptérů) je možno logicky sdružovat a vytvářet univerzální moduly.

Integrační cesta mezi aplikacemi a integrační platformou může sdružovat více integračních případů stejného typu z hlediska technologie (např. integrační cesta k aplikaci 1 může být využívána pro zasílání zpráv o zákaznících stejně jako o jejich produktech). To vede k tomu, že vytvářená rozhraní mohou sdílet funkcionalitu pro více integračních případů - tato optimalizace je prací návrháře, architekta.

Vzhledem k provázanosti dokumentace integračního případu je vždy snadné dohledat provázanost API aplikací, integračních platformy a integračního případu. Návrh rozhraní s popsanými integračními případy poté slouží jako zadávací dokumentace pro vývojáře adaptérů, integrační platformy a obou aplikací. V případě složitějších procesů (např. proces podporovaný více aplikacemi) jsou jednotlivé operace (integrační případy) propojovány integračním procesem, který popisuje celý proces řízený zpravidla integrační platformou.

9.7 Od analýzy k návrhu

Poté, co jsme si představili základní kroky fáze analýzy i návrhu integračních projektů, podívejme se nyní, jakým způsobem spolupracují a jak se ovlivňují. [BUSSLER, 2003] uvádí tři základní přístupy k modelování integračních řešení:

1. Přístup seshora - dolu (top - down) - modelování začíná od modelování procesů a integračních bodů na konceptuální úrovni a pokračuje jejich zpřesňováním.
2. Přístup zezdola - nahoru (bottom - up) - modelování začíná přesnou specifikací integračních bodů na detailní úrovni a postupně se na jejich základě vytvářejí procesy.
3. Přístup založený na abstrakci - modelování začíná modelováním jednoho procesu, které pokračuje přesnou specifikací integračních bodů. Modelování dalších procesů zpětně redefinuje specifikace integračních bodů.

Přístup popisovaný v této kapitole **kombinuje výše uvedené přístupy, odlišuje se zavedením abstrakce integračního případu a neomezuje se pouze na fázi návrhu procesů**. Probíhá v následujících krocích:

1. Analýza integračních požadavků, především funkčních (nefunkční požadavky jsou považovány za doplňkové k jádru řešení). Verifikuje se vazba integračních procesů na podnikové procesy, zda neexistují slepá místa procesů. Výstupem jsou rozhraní na podnikové procesy implementované v podnikových aplikacích a specifikace funkcionality integračních procesů.
2. Analýza integračních procesů přístupem seshora - dolu. Výstupem je rozpad na integrační případy a rozhraní.
3. Návrh řešení integračních případů pomocí rozhodovacího stromu integračních postupů. Výstupem je přiřazení integračních přístupů a technologií k integračním případům.
4. Podle navržených integračních postupů se pokračuje analýzou integračních případů až na úroveň detailní specifikace (ta se odlišuje podle integračních postupů). Výstupem je detailní popis integračního procesu včetně integračních bodů.
5. Přístupem zezdola nahoru se navrhují rozhraní (dochází ke slučování integračních bodů), sdílené prvky a podprocesy integračních procesů. Výstupem je detailní návrh řešení na úrovni integračních komponent a rozhraní dle zvolených integračních technologií.

Explicitně shrnuto, analytická fáze je založená na abstrakci integračního procesu a případu, do designové fáze se přechází na úrovni integračního případu na integrační přístup a technologii. Při přechodu z analýzy do návrhu **dochází ke změně orientace**. Zatímco analýza vychází z integračních procesů, potažmo z podnikových procesů, specifikace návrhu je orientována napříč integračními procesy podle integračních přístupů a technologií, resp. integračních komponent (které reprezentují zvolené třídy integračních produktů). Tento přechod není transparentní, což je potenciálním zdrojem problémů při následné údržbě systémů. V důsledku to znamená, že drobná změna v podnikových procesech, která se snadno promítne do analytických výstupů, vede k rozsáhlé změně v použití integračních technologií (integrační případ se např. realizuje zcela jinou integrační cestou).

Analýza a návrh integračních případů je základním podkladem pro testovací příklady integrační platformy a integrovaných aplikací.

V [QUARTEL, 2004] je popsána fáze analýzy a návrhu, který se orientuje na návrh služeb v SOA. Definuje 4 fáze - specifikaci podnikových procesů a specifikaci, návrh a implementaci aplikačních služeb (podobný postup popisuje také [BIEBERSTEIN, 2005]). Poslední fázi pouze zmiňuje s tím, že je závislá na konkrétní platformě. Pro modelování používá ISDL (The Interaction Systems Design Language). Ve vztahu k této práci se jedná o kompatibilní metodický postup, který vychází rovněž z procesního přístupu a lze jej uplatnit pro integrační koncepci SOA.

[ERL, 2005] rovněž popisuje implementaci SOA. Stanovuje tři možné strategie implementace - top-down, bottom-up a agilní přístup aplikovaný na šest základních fází projektu - analýzu, design, vývoj, testování, nasazení a provozování. Veškeré snažení je

orientováno na vytvářené služby, možné způsoby a principy jejich orchestrace a na vazbu na podnikové procesy. Ve vztahu k této práci pokrývají popsané strategie a postupy oblast integračních úloh řešených prostřednictvím SOA a je vhodné je využít jako rozšíření integrační metodiky pro tuto oblast.

Uvedené odkazy na projektové fáze a postupy specifické pro koncepci SOA dokumentují skutečnost, že k samotným integračním koncepcím, přístupům, ale i technologiím a produktům často existují dílčí části metodiky (např. v podobě tzv. best practices - tedy nejlepších zkušeností), nejčastěji pro fázi návrhu a vývoje. Tato dílčí doporučení a praktiky lze přirozeně aplikovat v souladu s metodikou prezentovanou v této práci podle zvoleného integračního postupu. Metodika je v tomto směru otevřená pro efektivní využití konkrétních postupů, zejména ve fázi detailního návrhu a vývoje, kde zůstává nezávislá na zvoleném integračním postupu.

10. Integrační postupy

Pro řešení integrace aplikací uvnitř podniku neexistuje univerzální nástroj, který by byl schopen řešit všechny na něj kladené požadavky. Objevují se ale nové integrační postupy, a tak se možnosti integračních nástrojů postupně rozšiřují.

V této kapitole se budeme zabývat různými postupy, kterak je možné aplikace integrovat. Používaná terminologie není v literatuře jednotná a jednotlivé termíny nejsou přesně definovány. Na začátku kapitoly tedy nejdříve vymezím termíny tak, jak jsou v literatuře nejčastěji používány.

Integračních postupů existuje celá řada a v souvisejících přílohách si u každého postupu stručně popíšeme jeho podstatu. Cílem není pochopit detailně celý postup, ten je možno nastudovat v informačních zdrojích, na které odkazuji, ale umět si daný postup zařadit mezi ostatní a pochopit vzájemné souvislosti. Výstupem této kapitoly je mapa integračních postupů, v které je možné se rychle zorientovat a vyhledat alternativní způsoby řešení integračních úloh. Pokud má architekt, či designer povědomí jen o úzké skupině integračních postupů, pak zákonitě dochází k jejich aplikaci i v úlohách, kde by bylo efektivnější užít jiný postup. Je prakticky nemožné ovládnout všechny integrační postupy na expertní úrovni, proto je nutné vytvářet integrační týmy složené ze specialistů na různé skupiny postupů, nebo akceptovat méně efektivní způsob řešení integračních úloh omezenou skupinou integračních postupů.

10.1 Přehled integračních postupů v informačních zdrojích

Pokoušel jsem nalézt v informačních zdrojích podobnou mapu, několik autorů se o ni pokoušelo, ale buď se jednalo jen o určitý výsek z celé problematiky nebo zde chybělo základní pojítko, podle čeho autor mapu strukturoval. V tabulce 5 - Přehled integračních postupů v informačních zdrojích (zdroj: autor) uvádím přehled nalezených přehledů integračních postupů v informačních zdrojích.

Identifikace informačního zdroje	Pokryté integrační oblasti	Nalezené nedostatky
[BISHOP, 2003]	rozlišuje middleware podle integrace (5 kategorií) a aplikace (6 kategorií)	pouze oblast middleware, nekompletní výčet, nezachyceny vzájemné vazby
[BUSSLER, 2003]	členění na point-to-point, hub-and-spoke a procesní integraci a dále pak na funkční (ve smyslu transakční), datovou (ve smyslu virtuální databáze) a událostní (ve smyslu zasílání zpráv)	nekompletní výčet, nezachyceny vzájemné vazby
[DEVARAJ, 2004]	rozlišuje integrační přístupy na 5 úrovních - prezentační služby, podnikové služby, podnikové procesy, podnikové entity, podniková data	pouze vybrané postupy, chybí návaznost na integrační technologie
[DIAMONDCLUSTER, 2001]	členění na prezentační, datovou a API integraci	pouze základní přehled
[HAUBRICH, 2005]	vychází z [HOHPE, 2003b], doplňuje SOA	nekompletní výčet
[HOHPE], [HOHPE, 2003b]	popsáno 5 integračních přístupů (označeny jako integrační strategie)	nekompletní výčet, nestrukturovaný přístup k členění
[IMHOFF, 2004]	porovnání ETL, EAI a EII	pouze základní přehled

[JANDOŠ, 2004]	členění na procesní, aplikační, datový přístup, vyjmenovány základní přístupy (označeny jako technologie a prostředky), strukturované členění	nezachyceny vzájemné vazby
[JUŘEK, 2004]	členění na integraci datovou, obchodní logiky, UI, detailní popis vybraných integrační postupů	pouze vybrané postupy
[JUŘEK, 2005]	mapa technologií zachycující vztahy	mapa je nekompletní a nepřesná, chybí detailní popis
[KAYE, 2003]	popisuje vybrané postupy, pouze pro porovnání s webovými službami	chybí některé přístupy a nejsou popsány souvislosti
[KRAFZIG, 2004]	popisuje RPC, ORB, MOM, TP a AS jako komunikační middleware	pouze vybrané postupy, žádné členění
[LINTHICUM, 1999]	členění na integraci datovou, API, metod, UI detailní popis přístupů	chybí vzájemné vazby a některé postupy
[LINTHICUM, 2003]	členění na informačně, procesně, servisně a portálově orientovanou integraci, detailní popis přístupů (označeny jako typy middleware)	nestrukturovaný přístup k členění
[MARCUS, 2002]	u vybraných postupů porovnává tržní produkty	pouze vybrané postupy, nestrukturovaný přístup k členění
[MCCOY, 2003b]	srovnání integračních technologií dle jejich vyzrálosti a viditelnosti	nekompletní výčet, nestrukturovaný přístup k členění
[OVUM, 2001]	členění na datovou, objektovou a procesní integraci, porovnává integrační produkty	pouze základní přehled, nezabývá se integračními postupy hlouběji
[SEACORD, 2003]	vybrané postupy pro integraci legacy aplikací	pouze vybrané postupy
[SEI]	detailně popisuje vybrané integrační způsoby	pouze vybrané postupy, chybí struktura integračních způsobů
[SCHMELZER, 2003]	základní porovnání přístupů integrace na zakázku, na datové úrovni, EAI/B2B a SOA	pouze základní přehled
[STRŮVER, 2002]	komentuje navržená členění z jiných informačních zdrojů a nabízí výčet integračních technologií	pouze vybrané postupy, chybí struktura integračních způsobů
[SULLIVAN, 2003]	vybrané postupy související s EIP	pouze vybrané postupy
[TSE]	členění na integraci datovou, metod, UI (označeny jako integrační modely), strukturované členění	chybí některé přístupy a nejsou popsány souvislosti

tabulka 5 - Přehled integračních postupů v informačních zdrojích (zdroj: autor)

[MCCOY, 2003b] ze společnosti Gartner Group (novější vydání reportu se mi nepodařilo získat) popisuje křivku vyzrálosti technologií souvisejících s integrací ve vztahu k

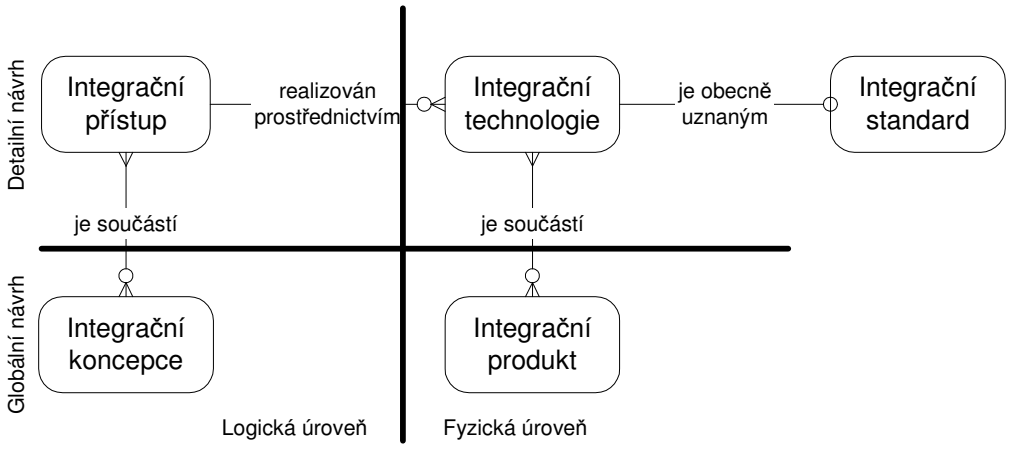
časové ose. K uvedeným technologiím uvádí i stručnou charakteristiku a hlavní dodavatele. Pomocí této křivky lze získat reálný přehled o aktuálnosti dané integrační technologie.

Za nejkompletněji a přehledně zpracované zdroje považuji [LINTHICUM, 2003], [LINTHICUM, 1999] a [JANDOŠ, 2004] a pro sestavení dále prezentované mapy integračních způsobů jsem je využil pro základní strukturu. O klasifikaci integračních koncepcí a technologií se pokusil rovněž [BUSSLER, 2002a]. Tuto klasifikaci nepovažuji pro potřeby této práce za vhodnou, neboť nestrukturovaně směšuje typy integrace, charakteristiku komunikace a rozhraní a charakteristiku vybraných technologií a ve výsledku používá tuto klasifikaci pouze pro tři vybrané integrační přístupy.

10.2 Definice integračních termínů

Oblast integrace aplikací obsahuje v současné době rozsáhlou **množinu integračních technologií, koncepcí, přístupů, produktů, standardů**. Souhrnně budeme tuto množinu označovat integračními postupy. **Integračním postupem tedy definujeme obecný postup nebo i jeho konkrétní implementaci, které jsou primárně určeny pro realizaci integrace aplikací**. Do integračních postupů zahrnujeme:

1. **Integrační přístupy** - definují způsob, jak integrovat aplikace za určitých podmínek, na logické úrovni.
2. **Integrační technologie** - jsou konkrétní implementací integračního přístupu na fyzické úrovni.
3. **Integrační koncepce** - tvoří kompaktní (synergickou) kombinaci několika integračních přístupů, které se uplatní současně v daném prostředí, na logické úrovni.
4. **Integrační produkty** - tvoří kompaktní kombinaci několika integračních technologií, které společně tvoří funkční celek na fyzické úrovni.
5. **Integrační standardy** - vybrané technologie, které jsou standardně implementovány v integračních produktech.



obrázek 19 - Základní vztahy mezi typy integračních postupů (zdroj: autor)

Integrační komponentou budeme označovat samostatný prvek informačního systému určený primárně pro integraci aplikací. **Integrační platformu** budeme v širším pojetí užívat jako obecný termín pro množinu integračních komponent implementovaných do PIS, v užším pojetí jako integrační komponentu v podobě integračního produktu typu BPI (Business Process Integration server).

V souvislosti s integrací se často setkáme s termínem **middleware**. Middleware definuje [DOHNAL, 1997] jako komponentu, která "zajišťuje transparentnost umístění distribuovaných prostředků, tedy že lze pracovat s distribuovanými zdroji jako kdyby byly lokální“.

[LOSAVIO, 2003] definuje EAI middleware jako "run time systémový software, který přímo umožňuje interakci (žádost o a odezvu na službu) napříč komponentami (nezávisle navrženými aplikacemi) v distribuovaném a síťovém prostředí".

[JANDOŠ, 2004] jej definuje následovně: "Middleware označuje množinu SW modulů s různou funkcionalitou, které

- zajišťují aplikačně – nezávislé služby (tj. nejsou orientovány na konkrétní aplikace),
- umožňují propojení a spolupráci (integraci a kooperaci) jednotlivých komponent SW architektury,
- jsou umístěny uprostřed (tj. in the middle) mezi základním SW (OS a základní komunikační služby – zajišťované vrstvami 1 až 3 RM OSI) a aplikačním SW."

10.3 Integrační koncepce

Integrační strategie definuje základní pravidla užití integračních přístupů v rámci jednoho podniku, zpravidla se opírá alespoň o jednu integrační koncepci, např. SOA. Je součástí informační strategie (viz kapitola 4.2 Informační strategie). Garantem dodržování integrační strategie je hlavní IT architekt podniku. Integrační strategie by měla pokrývat všechny obecné případy integrace aplikací, které podnik řeší. Její rozsah je závislý na vyzrálosti integrační úrovně podniku, kde rozeznáváme:

Integrační úroveň	Podstata integrace
Datová	Předání dat mezi aplikacemi.
Služeb (též aplikační)	Vykonání služby ve vzdálené aplikaci na pokyn volající aplikace.
Procesní	Realizace podnikového procesu mezi aplikacemi.
Prezentační	Konsolidace prezentační vrstvy více aplikací, agregace dat z více aplikací do jedné prezentační vrstvy.

tabulka 6 - Integrační úrovně (zdroj: autor)

Popišme si procesy probíhající na jednotlivých integračních úrovních:

1. Datová úroveň
    a. Volající aplikace (zdrojová nebo cílová aplikace) iniciuje přenos dat.
    b. Zdrojová aplikace připraví vybraná data.
    c. Data jsou vybraným způsobem přenesena k cílové aplikaci.
    d. Cílová aplikace může začít s daty pracovat. Mezi aplikacemi je třeba definovat oprávnění ke sdíleným datům (kdo se stará o referenční data), aby nevznikaly nekonzistence.
2. Úroveň služeb
    a. Volající aplikace zavolá službu volané aplikace a předá jí příslušné parametry (data).
    b. Volaná aplikace realizuje požadovanou službu.
    c. Pokud je vyžadováno, volaná aplikace předá volající aplikaci výsledek (data) provedené služby.
3. Procesní úroveň
    a. Volající aplikace vyvolá proces (zpravidla zasláním zprávy o nastálé události) volané aplikace a předá jí příslušná data.
    b. Volaná aplikace spustí příslušný proces, který je řízenou posloupností volání služeb dalších aplikací.
    c. Pokud je vyžadováno, volaná aplikace informuje volající aplikaci (případně i jiné aplikace) o průběhu (stavech) procesu až do jeho ukončení.
4. Prezentační úroveň
    a. Volající aplikace požádá volanou aplikaci o vytvoření vybrané obrazovky dle předaných parametrů (realizuje se např. pomocí URL adresy).

b. Volaná aplikace vyhotoví příslušnou obrazovku a pošle ji zpět cílové aplikaci.
c. Volající aplikace převezme buď celou obrazovku nebo její definované části (zejména data) a dále ji zpracuje. Cílem integrace nemusí být výsledná obrazovka, ale především operace provedené pro její zobrazení (např. uložení záznamu).

Vedle těchto úrovní se uvádí i úroveň objektová, která sjednocuje úroveň datovou a služeb. Podstatou objektové integrace je práce s objekty (tedy daty a metodami s nimi souvisejícími) mezi aplikacemi.

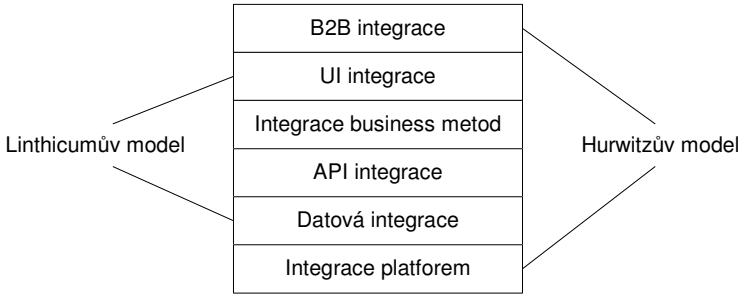
Informační zdroje nepoužívají jednotný pohled na integrační úrovně, [MOSAWI, 2006] uvádí a porovnává z různých informačních zdrojů 9 různých integračních přístupů (ve smyslu integračních úrovní) - datový, objektový, funkční/metod, uživatelského rozhraní, aplikačního rozhraní, prezentační, procesní, interně procesní, mezipodnikově procesní. Stejný zdroj rozlišuje integrační přístupy i podle vrstvy, s kterou pracují - zde z jiných zdrojů identifikuje vrstvu podnikové architektury, podnikových procesů, informační architektury, mezipodnikových procesů, aplikační architektury, podnikových aplikací, technologické architektury a middlewaru. V této práci se zabýváme primárně vrstvou technologické architektury a podnikových aplikací, neboť integrační projekt máme vymezený jako technologický projekt.

V literatuře se setkáme i s jiným členěním integračních úrovní - viz obrázek 20 - Úrovně integrace (zdroj: [SKOGLUND, 2002]). V těchto modelech se mísí dva odlišné pohledy - pohled technologický (neboli jak bude integrováno) a pohled předmětný (co bude integrováno). Konkrétně :

1. Integrace platforem - technologická integrace zaměřená na interoperabilitu platforem.
2. Datová integrace - lze ji chápat jako technologickou i předmětnou, více viz 17.1.2 Datová koncepce.
3. API integrace - technologická integrace využívající API aplikací.
4. Integrace business metod - předmětná integrace zaměřená na integraci interních procesů. Od API integrace se odlišuje tím, že poskytované služby (metody) nejsou poskytovány aplikacemi, ale existují nezávisle na aplikacích. Jsou uloženy na centrálním serveru nebo ve formě distribuovaných objektů v síti.
5. UI integrace - předmětná integrace integrující UI aplikací (není popsána ve smyslu technologické integrace).
6. B2B integrace - předmětná integrace zaměřená na integraci externích procesů.

V mém pojetí integračních úrovní jsem tyto dva pohledy oddělil - tabulka 6 - Integrační úrovně (zdroj: autor) definuje **předmětný pohled**, kapitola 10.4.1 Integrační rozhraní definuje **technologický pohled**. Integraci platforem a B2B integraci jsem se vzhledem k vymezené oblasti zkoumání dále nezabýval. Mezi API integraci a integraci business metod nestavím jasnou hranici a nazývám ji společně integrací služeb (z předmětného pohledu) a integrací skrze funkční rozhraní (z technologického pohledu). Ve vztahu k integračním potřebám jiných aplikací není podstatné, kdo službu poskytuje (zda aplikace či sdílená metoda) - tento přístup odpovídá SOA, a naopak lze platformu poskytující business metody považovat za rovnocennou podnikovou aplikaci.

Podobné členění používá i společnost Gartner (analyzováno v [HAUBRICH, 2005]), která rozlišuje integraci na zajištění datové konzistence (blíží se datové integraci), řešení vícekrokových procesů (odpovídá procesní integraci) a vytváření kompozitních aplikací. Kompozitní aplikace jsou aplikace, které jsou sestaveny ze služeb poskytovaných jinými aplikacemi nebo samostatně definovaných služeb (tzv. business metod v Linthicumově terminologii). Tento proces sestavování více služeb do jedné služby (nebo procesu, neboť i proces lze abstraktně chápat jako službu) se nazývá **orchestrace**.



obrázek 20 - Úrovně integrace (zdroj: [SKOGLUND, 2002])

Integrační strategie tedy říká, jaké integrační přístupy budou používány pro předávání dat mezi aplikacemi, jaké pro volání vzdálených služeb a jaké pro integraci podnikových procesů mezi aplikacemi. Předpřipravenými vzory pro integrační strategii jsou integrační koncepce, jejichž přehled je zpracován v příloze 17.1 Integrační koncepce.

10.4 Integrační přístupy

Konkrétní integrační přístup je obecnou šablonou, jak lze řešit integrační případ nebo jeho část. Z podstaty integračního případu řeší integrační přístup dvě základní věci (viz obrázek 21 - Komunikační schéma v integraci mezi aplikacemi (zdroj: autor)), resp. alespoň jednu z nich:

1. integrační rozhraní aplikace,
2. způsob komunikace mezi aplikacemi.

Integrační přístup budeme popisovat ve formě komunikačního modelu, který bude popisovat způsob, jak spolu aplikace komunikují.

obrázek 21 - Komunikační schéma v integraci mezi aplikacemi (zdroj: autor)

10.4.1 Integrační rozhraní

Integračním rozhraním označuji tu část aplikace, která je dedikována pro komunikaci s jinými aplikacemi. Někdy bývá nepřesně označováno jako API (Application Programming Interface), což je širší pojem, neboť zahrnuje i interní rozhraní určené pro vnitřní komunikaci aplikace. Dále se setkáme s pojmy adaptér a konektor. Adaptérem označujeme dle [IBM] samostatnou vrstvu mezi API a middlewarem pro zasílání zpráv. [JUŘEK, 2004] označuje adaptérem prvek spojující integrační broker s integrovanými aplikacemi. Konektorem označujeme dle [IBM] synchronní spojení na vzdálená rozhraní. Specifickým rozhraním je UI, či GUI (Graphical User Interface), které je určeno pro komunikaci s uživatelem aplikace.

Každá aplikace se skládá ze 3 základních vrstev (viz také [SEI], [FOWLER, 2002] (analyzuje i méně časté rozdělení do vrstev), [VOŘÍŠEK, 1997]) - viz obrázek 22 - Vrstvy aplikace s vnějšími rozhraními (zdroj: autor) (vrstva aplikační logiky se také nazývá vrstvou obchodní logiky nebo doménou [FOWLER, 2002]):

1. Datová - udržuje veškerá data využívané aplikací.
2. Funkční - podle pravidel implementované obchodní logiky pracuje s daty.
3. Presentační - zpracovává vstupy a výstupy aplikace ve spolupráci s funkční vrstvou.

Poznámka: Pro další zkoumání není relevantní, do jaké míry dodržuje aplikace striktní oddělitelnost těchto vrstev.

Terminologie pro označování vrstev není jednotná. [HOHMANN, 2003] používá ve stejném významu vrstvy označené jako:

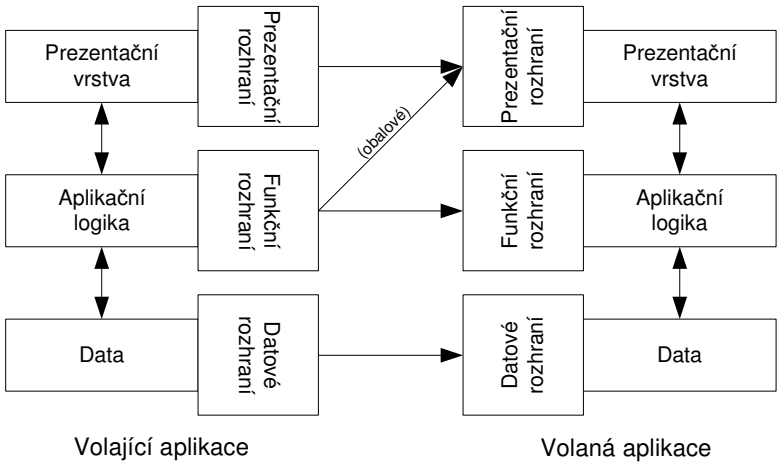
- uživatelské rozhraní (UI),
- služby (services),
- doménové objekty (domain objects),
- trvalá data (persistent data),

přičemž jejich význam je shodný. Pouze funkční vrstva je rozdělena do dvou - služeb (poskytují službu pro další aplikace a pro UI) a doménových objektů (volitelná vrstva vznikající při složitější logice aplikace, ve které jsou implementované vztahy mezi objekty aplikace a k nim příslušející business pravidla).

Osamocená aplikace si řeší všechny vrstvy samostatně bez jakékoliv vlivu zvenčí. Integrovaná aplikace má alespoň jednu z těchto vrstev sdílenou (ve smyslu, že ji využívá zvenčí a nebo ji naopak poskytuje).

Rozhraní rozlišujeme podle toho, ke které vrstvě aplikace přistupují, na:

1. Datová
2. Funkční
3. Prezentační



obrázek 22 - Vrstvy aplikace s vnějšími rozhraními (zdroj: autor)

Přiložená tabulka 7 - Možnosti rozhraní (zdroj: autor) upozorňuje na schopnost rozhraní podporovat různé integrační úrovně. Neplatí závislost, že funkční rozhraní automaticky znamená integrační úroveň služeb. Příkladem je funkční rozhraní, které poskytuje pouze funkce vracející data z databáze.

Integrační úroveň / Kategorie rozhraní	Datová	Služeb	Procesní	Prezentační
Datové	ano	ne	ne	ne
Funkční	ano	ano	ano	ne
Prezentační grafický	ne	ne	ne	ano
Prezentační obalový	ano	ano	ne	ne

tabulka 7 - Možnosti rozhraní (zdroj: autor)

10.4.1.1 Datová rozhraní

Datová rozhraní zpřístupňují data v **datovém úložišti**. Úložištěm dat je soubor dat v libovolném formátu v paměti (trvalé či dočasné) počítače. Sofistikovaným uložením dat je použití databáze. S rozvojem databázového konceptu vzniklo několik typů databází:

- relační,
- objektové,
- multidimenzionální.

K datům v databázi se přistupuje přes databázový middleware, který se používá pro lokální i vzdálený přístup. Rozeznáváme tato datová rozhraní:

1. Souborové - poskytují přístup přímo k souborovému systému.
2. Databázové - poskytují přístup do databázových struktur skrze databázový middleware.

10.4.1.2 Funkční rozhraní

Funkční rozhraní zpřístupňují služby aplikace. Tyto služby mohou být dostupné ve formě procedur a funkcí nebo ve formě objektů. Objekty jsou data a k nim příslušející služby. Funkční rozhraní dělíme na:

1. Čistě funkční - poskytují přístup k funkcím či procedurám aplikace. Podle jejich povahy se dále rozlišují na:
    a. Datové - jejich účelem je poskytnout požadovaná data skrze definované funkční rozhraní bez vazby na související business pravidla. Ačkoli by data mohla být poskytnuta i přes datové rozhraní, využívá se pro přístup k nim funkční rozhraní z důvodu:
        i. nekompatibility datových a funkčních rozhraní (pokud řeším integrační případ přes funkční rozhraní, nemohu k nim jednoduše přidat jedno datové rozhraní),
        ii. funkční rozhraní představují bezpečnější (vůči chybné operaci) přístup k datům než datová rozhraní (data jsou chráněna datovou i elementární aplikační vrstvou),
        iii. skrytí datové vrstvy před ostatními aplikacemi.
    b. Vykonávající služby - poskytují přístup ke službám s implementovanými business pravidly (např. služba založení nového zákazníka, kterému jsou automaticky přiděleny základní produkty).
2. Objektové - poskytují přístup k objektům aplikace.

Pokud aplikace neposkytuje nativně přístup ke svým službám, tak se provede tzv. **obalení** aplikace (wrapper) vnějším rozhraním, které navenek emuluje standardní funkční rozhraní a interně vůči aplikaci transformuje vnější volání do možností realizace služeb samotné aplikace. Detailně o tomto přístupu píše [SEACORD, 2003] a [STRÜVER, 2002].

10.4.1.3 Prezentační rozhraní

Prezentační rozhraní zpřístupňuje možnosti uživatelského rozhraní aplikace (UI). Rozlišujeme prezentační rozhraní:

1. Obalové - rozhraní transformuje uživatelské rozhraní do systémového rozhraní pro ostatní aplikace. Využívá se v případech, kdy není možné přistupovat k potřebným funkcím aplikace prostřednictvím funkčního či datového rozhraní, ale tyto funkce jsou dostupné přes uživatelské rozhraní. Vůči ostatním aplikacím se pak rozhraní chová jako obalové funkční rozhraní.

    Podle možností rozhraní se v [LINTHICUM, 1999] dále dělí na:

a. statické - na statických obrazovkách se vyhledává pomocí souřadnicového systému
b. dynamické - obrazovky jsou zpracovávány s využitím vyhledávání

2. Grafické - aplikace na základě vstupů z rozhraní vytvoří obrazovku nebo vybraný výsek obrazovky a předá jej volající aplikaci, která jej dále nezpracovává, ale přímo jej vloží do své prezentační vrstvy. Předmětem této integrace jsou tedy i formátovací údaje pro prezentační vrstvu, kterým musí volající aplikace rozumět.

10.4.2 Komunikace

Při integraci mezi aplikacemi se vzájemné komunikace účastní alespoň dvě rozhraní. Rozhraní 1 označíme za volající, neboť komunikaci iniciuje, a rozhraní 2 za volané, neboť na iniciaci odpovídá. Podle toho budeme používat i termíny volající a volaná aplikace, alternativně klient a server. U datových rozhraní se spíše používá pojmů zdrojová a cílová aplikace, podle toho, která aplikace poskytuje komunikovaná data. Ale i zde platí, že komunikaci musí jedna z aplikací iniciovat.

Obě rozhraní si musí navzájem rozumět, jinak bude komunikace neúspěšná. Předpokladem pro vzájemné porozumění je shoda syntaxe a sémantiky předávaných informací. Za tímto účelem byla vyvinuta celá řada komunikačních protokolů a jazyků, kterými se ale nebude zabývat, neboť jsou pro smysl této práce nevýznamné.

Pro zajištění těchto předpokladů lze v integraci použít **zprostředkovatele**. Ten přímou komunikaci rozdělí na dvě samostatně probíhající komunikace a funguje jako překladatel mezi aplikacemi. Jeho další úlohou je řízení komunikace, kde na žádost aplikace 1 vyhledá vhodnou aplikaci 2 a následně monitoruje či řídí probíhající komunikaci. V některých případech může i probíhající komunikaci doplňovat svými vlastními informacemi.

Komunikaci tak rozlišujeme na:

1. přímou - aplikace 1 komunikuje přímo s aplikací 2
2. zprostředkovanou - komunikace probíhá přes zprostředkovatele

10.4.2.1 Přímá komunikace

Přímou komunikaci charakterizujeme podle (viz také [LINTHICUM, 2003] a [BUSSLER, 2003]):

1. Určení volané aplikace
   - Point-to-point - komunikace probíhá pouze mezi dvěma předem známými aplikacemi.
   - Publish-subscribe - zpráva je zveřejněna všem přihlášeným zájemcům podle téma zprávy.
   - Broadcast-multicast - zpráva je odeslána všem dostupným aplikacím (broadcast) nebo odesílatelem vybraným aplikacím (multicast).
2. Čekání na odezvu
   - Synchronní - volající aplikace čeká na odpověď (request-response).
   - Asynchronní - volající aplikace pošle zprávu a nečeká na odpověď (send-and-forget).
3. Udržování spojení
   - Souvislá (connection-oriented) - během komunikace je udržováno spojení.
   - Nesouvislá (connectionless) - spojení není udržováno.
4. Frontování požadavku
   - Bez fronty - příchozí požadavky jsou zpracovány ihned nebo zahozeny.
   - S frontou - příchozí požadavky jsou zařazovány do fronty a zpracovávány postupně.

Přímá komunikace musí probíhat mezi rozhraními shodné kategorie (nejsou známy technologie umožňující jejich kombinování). Přehled integračních přístupů pro přímou komunikaci je zpracován v příloze 17.2.1 Přímá komunikace.

10.4.2.2 Zprostředkovaná komunikace

Zprostředkovanou komunikaci chápeme jako komunikaci minimálně tří aplikací, kde prostřední aplikace vystupuje v roli zprostředkovatele. Komunikace je řízena zprostředkovatelem (broker), který je schopen řídit komunikaci mezi více aplikacemi. Zprostředkovatel zajišťuje zejména směrování, řízení, monitorování a transformaci požadavků a poskytuje sdílené integrační komponenty.

Otázka integrace aplikace je pak redukována na rozhraní mezi aplikací a zprostředkovatelem (přičemž zprostředkovatel poskytuje jednotné API). Komunikace mezi jednotlivými aplikacemi má všechny charakteristiky přímé komunikace, a proto se zde zaměříme pouze na samotné zprostředkovatele. Přehled integračních přístupů pro zprostředkovanou komunikaci je v příloze 17.2.2 Nepřímá komunikace.

Použití zprostředkovatele při integraci neznamená automatický přechod k centralizované integraci. Proto jsem sestavil tabulku 8 - Centralizace zprostředkované komunikace (zdroj: autor), kde je křížkem ke každému zmíněnému modelu komunikace označeno, zda je primárně používán jako centralizovaný nebo decentralizovaný. Centralizovaně zaměřený znamená správu sdílených dat, komponent či služeb z jednoho místa pro více aplikací. Tyto sdílené prvky však mohou být distribuovány a replikovány dle potřeby. Decentralizovaně zaměřený znamená, že se předpokládá koexistence více samostatných nezávislých instancí modelu bez vzájemných spojitostí.

Komunikační model	**Decentralizovaně zaměřený**	**Centralizovaně zaměřený**
Replikace a transformace	X	
Datový sklad		X
Virtuální databáze		X
MOM	X	
MB		X
ORB		X
SB		X
TP		X
AS		X
BPI		X
Webový server		X
EIP		X
WB	X	
Snímání obrazovek	X	

tabulka 8 - Centralizace zprostředkované komunikace (zdroj: autor)

10.5 Integrační technologie

Integrační technologie je konkrétní realizací (implementací) integračního přístupu. Často se setkáme s tím, že z jednoho integračního přístupu vznikla právě jedna integrační technologie. Někdy jich vznikne více, což je způsobeno odlišným detailním zpracováním přístupu do technologie. V důsledku toto rozštěpení může vést k odlišným standardům pro řešení stejného integračního problému a vzájemné nekompatibilitě. Naopak některé přístupy se postupně sloučili do jedné kompaktní technologie pro získání synergickému efektu.

Přehled integračních technologií, který vychází z navržené struktury integračních přístupů ve formě komunikačních modelů, je uveden v příloze 17.3 Integrační technologie.

10.6 Přehled integračních postupů

Tabulka 9 - Přehled integračních postupů (zdroj: autor) ukazuje uspořádané integrační postupy, které se mi podařilo identifikovat v informačních zdrojích a při praktickém zkoumání. Oproti přehledům, které jsou zmíněny v kapitole 10.1 Přehled integračních postupů v informačních zdrojích, je tento přehled

- Jasně strukturovaný (dle kategorie rozhraní a typů komunikace).
- Založený na samém principu integrace aplikací - tedy existenci rozhraní aplikací a jejich vzájemné komunikaci.
- Odhaluje souvislosti mezi integračními postupy - z tabulky např. vyčteme, že pro integraci skrze funkční rozhraní se pro přímou komunikaci používají 3 základní přístupy a pro zprostředkovanou komunikaci 7 přístupů. Dále z tabulky vidíme, že integrační přístup zasílání zpráv se při zprostředkované komunikaci rozvíjí do dvou integračních přístupů - MB a MOM.
- Popsané technologie a přístupy jsou poplatné době vzniku této práce. Cílem tohoto přehledu není primárně poskytnout aktuální přehled všech integračních technologií, ale získat elementární přehled o rozsahu používaných technologií a navrhnout základní kostru jejich katalogizace. **Pro metodiku jsou tedy podstatná kritéria členění dle kategorie rozhraní a dle typu komunikace, zmíněné přístupy a technologie pak dokumentují schopnost tohoto členění obsáhnout celou oblast**.
- Dostatečně flexibilní a otevřený, aby byl schopen pojmout nově vytvářené integrační technologie a přístupy.
- Obsahuje několik nových názvů integračních postupů, které byly identifikovány díky systematickému přístupu (např. rozdělení prezentačních rozhraní na grafické a obalové).

Technologie přímé komunikace	**Přístupy přímé komunikace**	**Kategorie rozhraní**	**Přístupy zprostředkované komunikace**	**Technologie zprostředkované komunikace**
Datové				
Souborové	Replikace		Replikace a transformace	ETL
				Transformace na zakázku
Databázové			Datový sklad	ODS
				DWH
Databázový link	Databázový link		Virtuální databáze	SQL gateway
				XQuery gateway
Funkční				
CORBA RPC	RPC		ORB	COM+/DCOM
Java RMI, EJB				CORBA
.NET Remoting				Java/RMI & JNDI
PL/SQL procedura				
Webová služba			SB	SB
CORBA Messaging	Zasílání zpráv		MOM	MOM
JMS				
MSMQ			MB	MB
Webová služba				
CORBA Transaction	Konverzační		TP	TP
			AS	založený na .NET

JTS				založený na J2EE
				založený na CORBA
.NET Transaction			BPI	Integrační hub
				Integrační sběrnice
Prezentační grafické				
ASP	Žádost o stránku		Webový server	Webový server
JSP			EIP	EIP (portlety)
PHP			Webový prohlížeč	Webový prohlížeč (rámce)
Prezentační obalové				
n/a	n/a		Snímání obrazovek	analýza zdrojového kódu
				OCR
				rozeznávání objektů

tabulka 9 - Přehled integračních postupů (zdroj: autor)

Pozn.:

- Odborné názvy některých přístupů či technologií (např. XQuery Gateway) nejsou dosud dostatečně ustálené, proto je potřeba v případě nejistoty o významu daného názvu vyhledat popis příslušného integračního postupu v příloze.
- Sestavení tohoto přehledu bylo věnováno úsilí přiměřené nastaveným cílům tohoto přehledu. Z tohoto důvodu mohou existovat integrační přístupy a technologie, které nejsou v přehledu zařazeny. Tento přehled zachycuje podstatnou část integračních technologií a přístupů a je dostatečně vypovídající.

10.7 Rozhodovací strom integračních postupů

Do schématu na obrázku 23 - Rozhodovací strom integračních postupů jsem sestavil rozhodovací strom pro výběr vhodného integračního postupu. Body D01-D22 jsou místa, kde se výběr integračního způsobu pro konkrétní integrační případ rozhoduje podle kritérií uvedených v tabulce 10 - Rozhodovací body stromu integračních postupů (zdroj: autor). Rozhodovací strom je navržen pro modelovou situaci, kde výběr není ovlivněn externími okolnostmi (např. vlastnostmi konkrétního integračního produktu, chybějícími experty pro daný integrační postup, atypickými faktory integračního případu). Při sestavování tohoto stromu jsem vycházel z praktických zkušeností a z informací dostupných v informačních zdrojích.

Rozhodovací strom dodržuje logiku prezentovanou v této kapitole, a to rozdělením rozhodovacího stromu do 5 úrovní (jsou v patrné ze schématu). Směrem od integračního případu k integračním přístupům a technologiím je strom více závislý na aktuálně známých integračních postupech. To znamená, že s rozvojem nových integračních postupů se bude základna stromu nejvíce rozšiřovat odspodu.

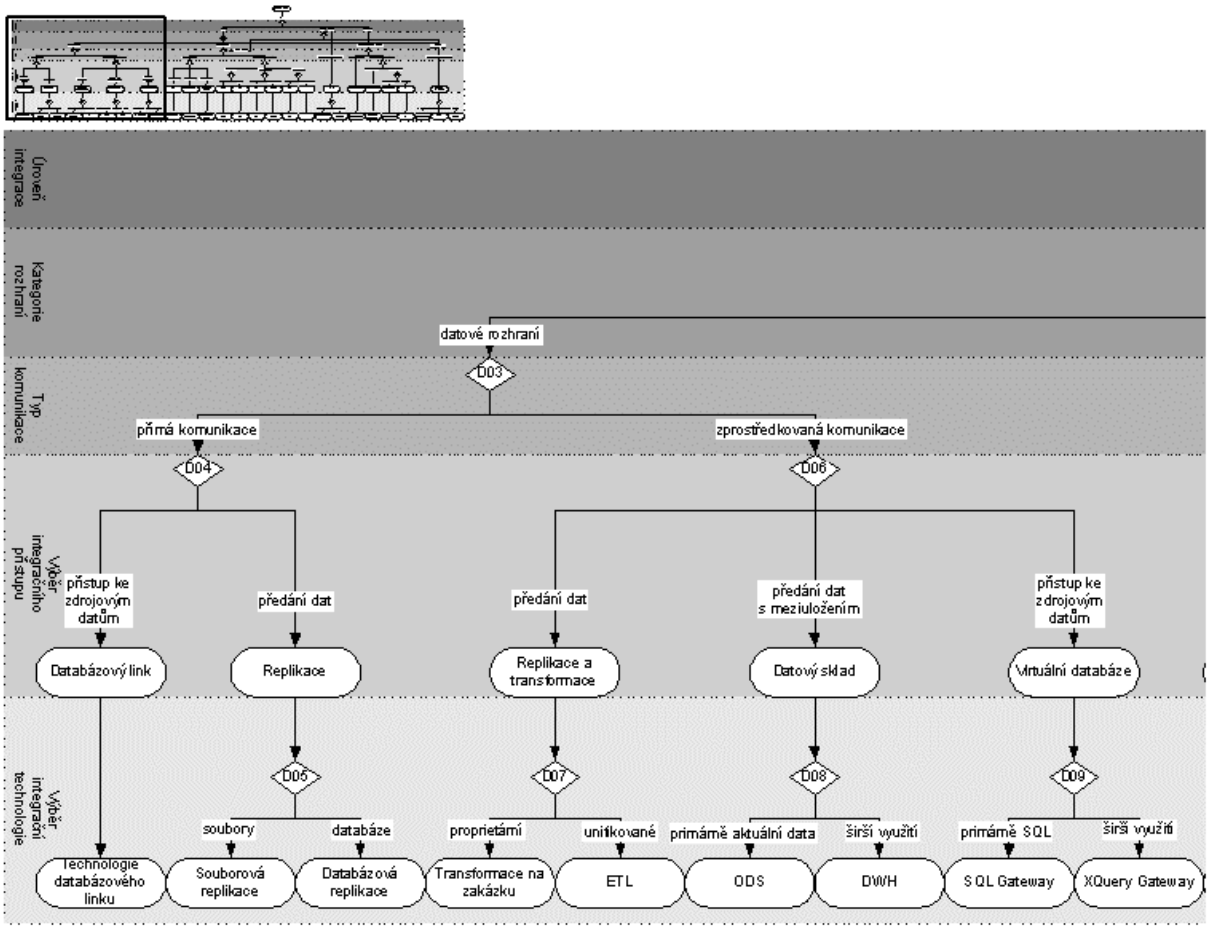
Rozhodovací body z horních třech úrovní označuji za středně až dlouhodobě stabilní body, neboť pokrývají kompletní množinu možných variant, např. úroveň kategorie rozhraní vychází z dlouhodobého členění vrstev aplikací. Totéž bychom mohli říci i o některých rozhodovacích bodech na čtvrté úrovni.

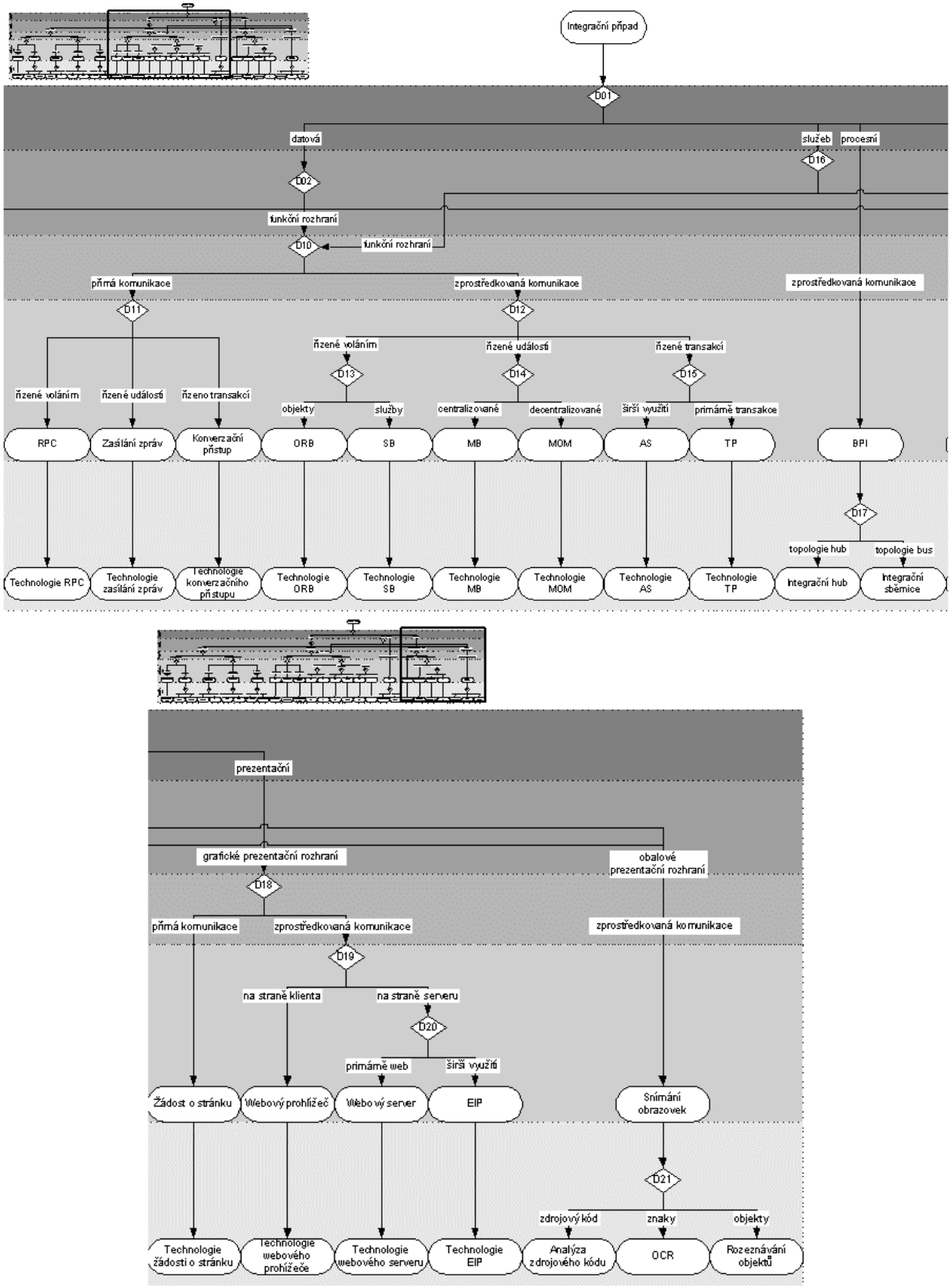
1. úroveň rozhodování (D01) - rozhodnutí o integrační úrovni,
2. úroveň rozhodování (D02, D17, u zbývajících větví není třeba) - rozhodnutí o kategorii rozhraní,
3. úroveň rozhodování (D03, D11, D19) - rozhodnutí o typu komunikace

4. a další úrovně rozhodování - rozhodnutí mezi možnými integračními přístupy a na dalších úrovních o možných integračních technologiích.

Pozn.:

- U některých technologií jsem rozhodovací bod nedefinoval, neboť jej nelze obecně stanovit. Tento strom je určen pro základní orientaci v aplikaci integračních postupů.
- Integrační strom se uplatňuje při plánování rozvoje integrační platformy (integrační strategie), kdy lze po vytipování klíčových integračních případů vybrat nejvhodnější přístup či technologii pro rozšíření integrační platformy.
  Ve fázi realizace projektu se uplatňuje rovněž, ale odlišným způsobem. Zde je potřeba nejdříve zaslepit ty větve rozhodovacího stromu, které vedou k postupům, které nejsou podporované integrační platformou, a u konkrétních případů ty větve, které odkazují na rozhraní nepodporované integrovanými aplikacemi.
- Mnohé rozhodovací body jsou závislé na možnostech integrovaných aplikací, např. v bodu D02 podle typu rozhraní, které aplikace podporují. Je-li na výběr více možností, pak se rozhodování řídí sekundárními kritérii, které označíme jako **ekonomicko-koncepční faktory**.
  Těmito faktory míníme ve vztahu k uvažované variantě kombinaci nákladů na realizaci, míru uspokojení požadavků vyplývajících z integračního případu a soulad s integrační strategií, je-li některá z variant v integrační strategii preferována.





obrázek 23 - Rozhodovací strom integračních postupů (3 části) (zdroj: autor)

Rozho-dovací bod	Popis	Výsledek rozhodnutí
D01	Jaká je požadovaná úroveň integrace? Kritériem je podstata integračního případu (viz tabulka 6 - Integrační úrovně (zdroj: autor)).	Zvolená úroveň integrace.
D02	Přes jaký typ rozhraní budou aplikace komunikovat? Množinu povolených kombinací definuje tabulka 7 - Možnosti rozhraní (zdroj: autor). Kritériem je množina typů rozhraní, přes jaká je aplikace schopna komunikovat. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolená kategorie rozhraní integrovaných aplikací.
D03	Bude komunikace přímá nebo zprostředkovaná? Primárním kritériem je návratnost investice do zprostředkovatele komunikace. Ta se zvyšuje s počtem integračních případů, které využívají stejného zprostředkovatele, a s přínosy samotného zprostředkovatele (řešení komplikovaných integračních případů, monitorovací a administrativní schopnosti zprostředkovatele, snazší údržba). Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolený typ komunikace.
D04	Jakým způsobem mají být data dostupná cílové aplikaci? Kritériem je potřeba cílové aplikace přistupovat k datům přímo do zdrojové aplikace nebo data dostávat k vlastnímu zpracování.	Zvolený integrační přístup pro přímou komunikaci skrze datové rozhraní.
D05	Jakým způsobem budou data replikována? Kritériem je schopnost aplikace poskytovat data ve formě souborů nebo databázových exportů či objektů. Preferovanou variantou je integrace na databázové úrovni. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolená integrační technologie pro replikaci.
D06	viz D04, rozšířený o variantu předávání dat s trvalejším uložením dat i ve zprostředkovateli integrace. Tato varianta se použije v případě, že je žádoucí omezit zatěžování zdrojového systému integrací s dalšími systémy a zároveň zátěž generovaná dotazy cílových systémů za jednotku času je vyšší než zátěž způsobená aktualizovaných dat zasílaných ze zdrojového systému do zprostředkovatele.	Zvolený integrační přístup pro zprostředkovanou komunikaci skrze datové rozhraní.
D07	Bude integrace proprietární nebo unifikovaná pomocí ETL? Primárním kritériem je návratnost investice do unifikovaného řešení dle ekonomicko-koncepčních faktorů.	Zvolená integrační technologie pro replikaci a transformaci.
D08	Potřebujeme datový sklad charakterizovaný jako ODS nebo DWH? Primárním kritériem jsou potřeby integračního případu k možnostem ODS a DWH, zejména zda se bude pracovat s historickými záznamy, které jsou typické pro DWH.	Zvolená integrační technologie pro datový sklad.

D09	Preferuje modernější způsob dotazování přes XQuery, který nám umožní přistupovat i do XML dokumentů, nebo nám stačí SQL? Primárním kritériem je předpokládané využití datových zdrojů, k nimž je nutno přistupovat primárně přes XQuery.	Zvolená integrační technologie pro virtuální databázi.
D10	viz D03	Zvolený typ komunikace.
D11	Bude integrační případ realizovaný voláním služeb (RPC), zasíláním zpráv nebo jako součást transakce mezi aplikacemi? Primárním kritériem je způsob iniciace integrace - potřebou volající aplikace, která zavolá službu, nebo událostí, kterou je třeba předat ve formě zprávy cílovým aplikacím, nebo jako důsledek volání jiné služby či odeslané zprávy? Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolený integrační přístup pro přímou komunikaci skrze funkční rozhraní.
D12	viz D11	Zvolená skupina integračních přístupů pro zprostředkovanou komunikaci skrze funkční rozhraní.
D13	Budou aplikacemi integrovány prostřednictvím objektů nebo služeb? Primárním kritériem je způsob integrační technologie podporovaný aplikacemi - jsou-li na rozhraní vystaveny objekty nebo služby. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolený integrační přístup pro zprostředkovanou komunikaci skrze funkční rozhraní.
D14	Využije se centralizovaný či decentralizovaný přístup? Primárním kritériem jsou požadavky integračního případu na centralizované řešení (např. centralizovaná správa), které stojí oproti riziku, že se centrální bod řešení stane zároveň úzkým místem. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolený integrační přístup pro zprostředkovanou komunikaci skrze funkční rozhraní.
D15	Postačí pro vytváření integrované funkční vrstvy TP nebo se využije AS? Primární kritériem je, zda smyslem integračního případu je pouze realizace transakcí nebo zda vyžaduje i vytvoření centralizovaného frameworku pro funkční vrstvu, jaký poskytují produkty AS. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolený integrační přístup pro zprostředkovanou komunikaci skrze funkční rozhraní.
D16	viz D02 bez možnosti datového rozhraní	Zvolená kategorie rozhraní integrovaných aplikací.
D17	Jaká je preferovaná topologie integrační platformy - hub (centralizovaná), bus (decentralizovaná) - viz D15? I v topologii bus je možnost implementace decentralizovaného prvku, který zajišťuje centralizovanou správu ostatních prvků. Z toho důvodu	Zvolená integrační technologie pro BPI.

	je třeba často použít pro rozhodnutí ekonomicko-koncepční faktory.	
D18	viz D03	Zvolený typ komunikace.
D19	Je preferována integrace prezentační vrstvy na straně klienta nebo serveru? Primárním kritériem je kooperace mezi poskytovali obsahu pro prezentační vrstvu. Je-li mezi nimi úzká spolupráce s cílem vytvářet kompaktní prezentační vrstvu, pak je třeba směřovat k řešení na straně serveru, naopak na straně webového klienta je odpovědnost za části sestavované prezentační vrstvy delegována na poskytovatele a části nejsou zpravidla úzce provázány. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolená skupina integračních přístupů pro zprostředkovanou komunikaci skrze prezentační grafické rozhraní.
D20	Postačí pro vytváření integrované prezentační vrstvy webový server nebo se využije EIP? Primární kritériem je, zda smyslem integračního případu je pouze sjednocování prezentační vrstvy, nebo zda vyžaduje i vytvoření centralizovaného frameworku pro prezentační vrstvu, jaký poskytují produkty EIP. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolený integrační přístup pro zprostředkovanou komunikaci skrze prezentační grafické rozhraní.
D21	Jakým způsobem bude možné vytvořit obalové rozhraní nad snímanými obrazovkami? Primárním kritériem jsou možnosti rozpoznávání integračních prvků pro rozhraní na snímaných obrazovkách - skrze analýzu zdrojového kódu, rozpoznávání znaků (OCR), či objektů. Sekundárním kritériem jsou ekonomicko-koncepční faktory.	Zvolená integrační technologie pro snímání obrazovek.

tabulka 10 - Rozhodovací body stromu integračních postupů (zdroj: autor)

10.8 Integrační produkty

Přehled integračních produktů byl vyčleněn do samostatné přílohy 17.4 Integrační produkty, neboť sám o sobě není součástí metodiky. Jeho smyslem je dokumentování vztahu metodiky, přesněji její části zabývající se integračními postupy, k aktuálnímu stav na trhu s těmito produkty.

10.9 Integrační standardy

Standardy užívanými při integraci se nebudeme detailněji zabývat, neboť jejich kompletní výčet by neměl žádný přínos k této práci. Podstatné standardy byly zmíněny v předchozích kapitolách a souvisejících přílohách a není třeba se k nim vracet. Za nejdůležitější standardizaci v oblasti integrace podnikových aplikací považuji množinu standardů, které se vážou k webovým službám a množinu standardů J2EE. Vedle obecně uznaných standardů se podobně jako v jiným oblastech vyvinulo množství de facto standardů, které prosadil dominující dodavatel pro daný tržní segment. Přehledově se integračním standardům (včetně B2B) věnuje např. [BUSSLER, 2003].

11. Model komplexní integrované architektury PIS

V předchozí kapitole 10 Integrační postupy a souvisejících přílohách jsme si představili používané integračními postupy. Každý z těchto postupů řeší definovanou skupinu integračních problémů. Aby se postup prosadil a začaly se pro něj vytvářet produkty, které budou podniky kupovat a používat, musí být skupina řešených problémů co možná nejširší a, pokud možno, řešit integrační problém, který stávající postupy nebyly schopny uspokojivě vyřešit. Za tímto účelem je každý nově vytvořený postup (zejména koncepce) prezentován jako všeobjímající a skoro vše řešící. V této kapitole se podíváme na tyto postupy ze strany podniku, který si vybírá svou integrační koncepci a stanovuje integrační strategii.

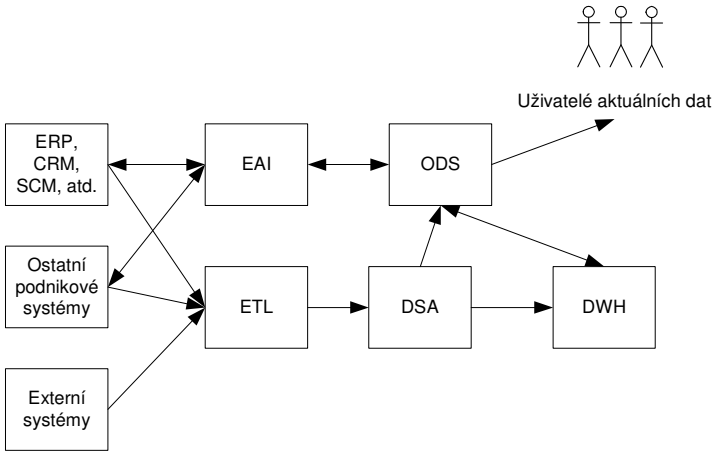
Přestože by se mohlo zdát, že datová integrace je v současné době již překonána vyššími stupni integrace, není tomu tak. Stále má své oprávnění v situacích, kdy je třeba předávat on-line velké množství dat, a v případech, kdy je výhodnější dávkové zpracování dat. V obou případech se k datové integraci přistupuje zejména z důvodu požadovaného výkonu. V závislosti na konkrétních případech může být datová integrace levnější oproti jiným přístupům.

11.1 Integrační modely PIS v informačních zdrojích

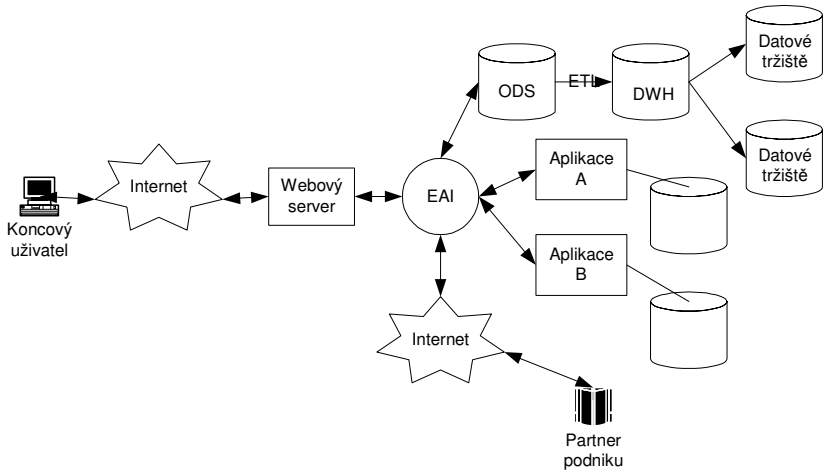
Přestože jsem prostudoval velké množství informačních zdrojů o integraci aplikací, nenalezl jsem komplexní model, který by kombinoval základní integrační koncepce a přístupy a vytvářel tak kompletní rámec pro jejich efektivní spolupráci.

Identifikace informačního zdroje	Pokrytá oblast
[NOVOTNÝ, 2005]	Koncept ODS coby jednotného místa datové integrace aktuálních dat z primárních systémů - viz obrázek 24 - Koncept ODS (zdroj: [NOVOTNÝ, 2005]. Kombinuje ODS, ETL, DWH a EAI, primárně za účelem plnění dat do DWH a ODS.
[ALAM, 2001]	Řeší umístění EAI vůči aplikacím s uživatelským rozhraním. Obrázek 25 - Forward integrační model (zdroj: [ALAM, 2001]) zachycuje tzv. Přední integrační model, který se používá v případech, kdy akce koncového uživatele spouští transakci protékající více aplikacemi. Požadavek je tedy z uživatelského rozhraní předán přímo do EAI nástroje, která transakci řeší s příslušnými aplikacemi. Obrázek 26 - Back-End integrační model (zdroj: [ALAM, 2001]) zobrazuje tzv. Zadní integrační model, který se používá v případě, kdy transakci spuštěnou uživatelem realizuje pouze jedna příslušná aplikace. Požadavek z uživatelského rozhraní je tedy přesměrován přímo do příslušné aplikace, která transakci zpracuje. EAI nástroj do procesu může zasáhnout v případě, že je aplikací požádán o provedení určité operace nebo v databázi aplikace nastane definovaná událost. Obrázek 27 - Hybridní integrační model (zdroj: [ALAM, 2001]) je kombinací obou výše popsaných modelů. Požadavek uživatele je dle daných pravidel předán buď do EAI nástroje, nebo přímo do vybrané aplikace k zpracování.
[PANCHA, 2002]	Pojednává o koexistenci ETL a ODS s EAI s důrazem na užití ETL a ODS pro dávkové operace a EAI pro transakce.
[DEEB, 2005]	Doporučuje kombinaci ETL a EAI. Upozorňuje na to, že samotné ETL neodkáže reagovat v reálném čase a EAI nedokáže zpracovávat velké množství dat za jednotku času.

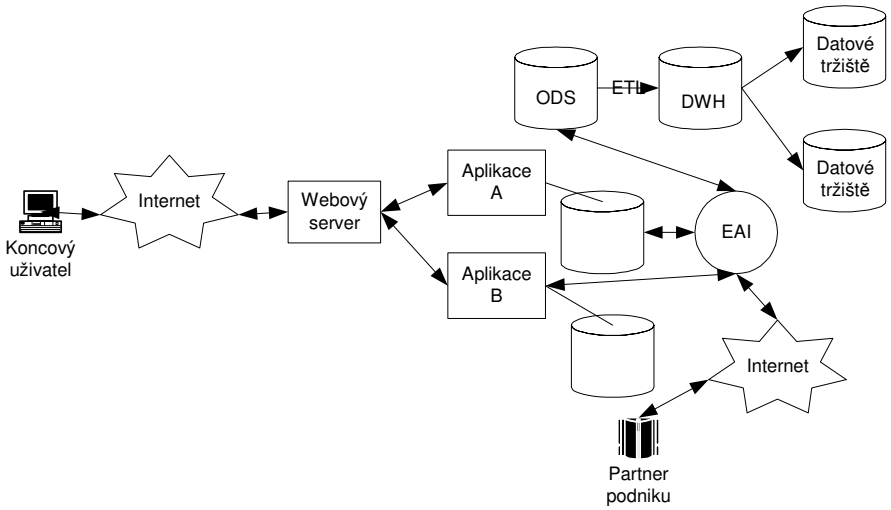
tabulka 11 - Přehled integračních modelů PIS v informačních zdrojích (zdroj: autor)



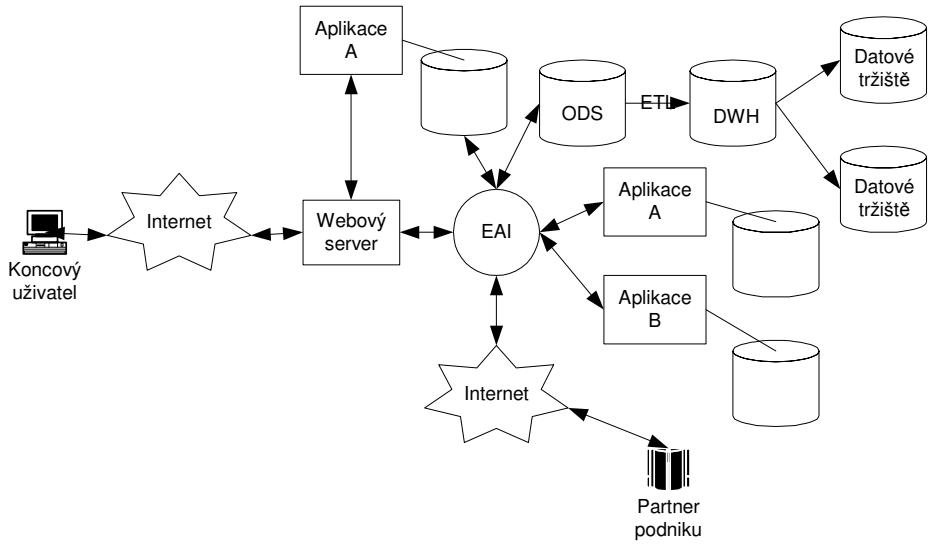
obrázek 24 - Koncept ODS (zdroj: [NOVOTNÝ, 2005])



obrázek 25 - Forward integrační model (zdroj: [ALAM, 2001])

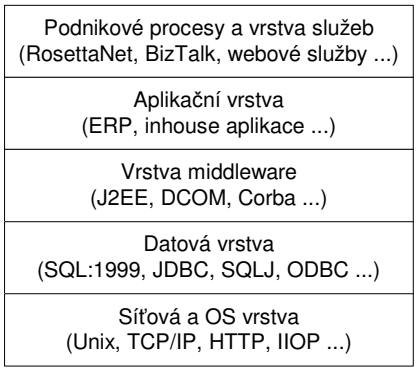


obrázek 26 - Back-End integrační model (zdroj: [ALAM, 2001])



obrázek 27 - Hybridní integrační model (zdroj: [ALAM, 2001])

Vedle těchto modelů jsem v informačních zdrojích nalezl model umístění integračních technologií ve vrstvách podnikového informačního systému jako celku. Obrázek 28 - Architektura PIS (zdroj: [FEUERLICHT, 2001]) ukazuje middleware jako rozhraní mezi daty a aplikacemi a nadřízenost podnikových procesů nad aplikační vrstvou.



obrázek 28 - Architektura PIS (zdroj: [FEUERLICHT, 2001])

V tabulce 12 - Rozšířený model OSI (zdroj: [GOLDSTONE]) nalezneme rozšířený model OSI (Open System Interconnection) pro EAI (popis základního modelu je v ([VOŘÍŠEK, 1997], str. 175)).

Vrstva	Název	Popis	Zdroj znalostí
12	Podnikový proces	Definuje specifické podnikové procesy	Provozní uživatelé
11	Podniková sémantika	Udržuje specifické podnikové datové definice a struktury	Provozní uživatelé a IS personál
10	Aplikační sémantika	Obsahuje znalost aplikační struktury a smyslu	Dodavatel aplikace
9	Syntax rozhraní	Definuje metody pro posílání a přijímání dat	Dodavatel aplikace
8	Middleware	Integrační architektura	Dodavatel middleware
7	Aplikační	Poskytuje standardizované služby	
6	Prezentační	Kóduje, šifruje a specifikuje formáty	

		přenášených dat	
5	Relační	Řídí protokoly spojení	
4	Transportní	Řídí síťové spojení a přenos packetů	
3	Síťová	Směruje packety	
2	Linková	Rozděluje packety a kontroluje fyzický datový tok	
1	Fyzická	Elektrické a mechanické specifikace	

tabulka 12 - Rozšířený model OSI (zdroj: [GOLDSTONE])

11.2 Komplexní integrační model PIS

Na základě praktických poznatků z integračních projektů, kterých jsem se účastnil (viz kapitola 17.6 Případové studie), a teoretických informací z informačních zdrojů, které jsou shrnuty v kapitole 10 Integrační postupy a souvisejících přílohách, jsem sestavil obrázek 29 - Model komplexní integrované architektury PIS (zdroj: autor). Toto schéma zachycuje:

- Hlavní integrační cesty v podnikovém informačním systému, tedy
  - kompletní sadu integračních cest mezi integračními komponentami PIS,
  - základní integrační cesty mezi podnikovými aplikacemi.
- Koexistenci stěžejních integračních komponent, resp. koncepcí a přístupů, současnosti v maximálně možné kombinaci, která umožňuje integrační případ řešit nejlepším možným postupem.

Integrační cestou označuji obecně sadu integračních případů, které probíhají mezi dvěma komponentami podnikového informačního systému. Jako příklad integrační cesty z modelu si uveďme propojení mezi podnikovým informačním portálem a integračním serverem. Přes toto spojení jsou reálně realizovány řádově stovky integračních případů, které se liší svým obsahem, ale probíhají právě mezi těmito dvěma komponentami. Díky zavedení této abstrakce, integrační cesty, můžeme popsat některé charakteristické rysy skupiny integračních případů, které cesta sdružuje, což nám pomůže v pochopení integračních procesů v podnikovém informačním systému.

Model nezachycuje:

- Integrační koncepci point-to-point, kterou lze aplikovat individuálně, a proto nemá význam ji v modelu zobrazovat.
- Integraci prostřednictvím snímání obrazovek, neboť i u ní se jedná o individuální řešení, které nemá význam v modelu zobrazovat.
- Alternativní integrační postupy, které jsou z koncepčního hlediska nepreferované (např. místo ETL lze použít i transformaci pomocí PL/SQL skriptů; ETL však nabízí širší a komfortnější možnosti, proto je preferován).

Poznámky k modelu:

- Směr šipek označuje převažující směr iniciace komunikace od volající k volané aplikaci či komponentě; u datových přenosů směr od zdrojové k cílové databázi.
- Přestože je ETL znázorněn jako jeden prvek v modelu, reálně se může jednat o několik jeho instancí.
- Popis k integračním komponentám v modelu (např. ETL) neuvádím, neboť je uveden v kapitole 12 Modely integračních nástrojů.
- Tečkovaná spojení označují alternativní integrační cesty, plná označují hlavní integrační cesty.
- ODS a DWH jsou integrační komponenty, jejichž obsah je obrazem zdrojových aplikací. Je tedy nutno respektovat posloupnost, že data se primárně aktualizují ve zdrojové aplikaci a následné (případně v transakci) se aktualizuje ODS a DWH.

- Model obsahuje integrační komponentu XQuery Gateway, avšak upozorňuji na to, že tato komponenta je relativně nová (první implementace v roce 2005) a neměl jsem možnost posoudit její vlastnosti v praxi.
- V modelu je jako jádro EAI uveden obecně integrační server. Pokud nemá podnikový informační systém problém přejít na SOA, tedy využívání primárně webových služeb pro komunikaci s aplikacemi, pak lze doporučit variantu BPI - ESB, ovšem s plnohodnotnou podporou procesů (nikoli jen tzv. odlehčenou variantu).
- Generování stránky může skrývat rovněž provedení akcí. Např. při přihlašování do aplikace je po odeslání přihlašovacích údajů zobrazena další stránka aplikace, na pozadí však proběhly operace ověřující zaslané přihlašovací údaje.

Tabulka 13 - Popis integračních cest v modelu (zdroj: autor) popisuje na elementární úrovni integrační cesty v modelu. Tyto integrační cesty můžeme pro snazší orientaci rozdělit na:

- interní, které probíhají mezi integračními komponentami, a
- externí, které probíhají mezi integračními komponentami a aplikacemi (a uživatelem skrze GUI).

Číslo cesty	Popis	Externí/ interní typ
01	Žádost webového prohlížeče o vygenerování stránky z volané aplikace. Poznámka: Komunikace probíhá technologicky přes vlastní webový server aplikace, který není znázorněn pro udržení přehlednosti modelu.	E
02	Žádost specifického UI aplikace o vygenerování stránky z volané aplikace.	E
03	Žádost webového serveru o vygenerování stránky z volané aplikace.	E
04	Žádost EIP o vygenerování stránky z volané aplikace.	E
05	Žádost BPI o provedení služby volanou aplikací.	E
06	Žádost volající aplikace o zprostředkování služby nebo procesu integračním serverem.	E
07	Žádost BPI o aktualizaci dat v ODS nebo jejich vyhledání.	I
08	Předání dávky dat ze zdrojové aplikace ke zpracování v ETL. Poznámka: Dočasné datové úložiště není povinné; zpravidla se užívá pouze v případě, pokud by prováděné datové operace zatížily neúměrně zdrojovou aplikaci.	E
09	Žádost EIP o provedení služby ve volané aplikaci.	E
10	Žádost EIP o aktualizaci dat v cílové aplikaci nebo jejich vyhledání.	E
11	Žádost ETL o aktualizaci dat v cílové aplikaci nebo jejich vyhledání pro potřeby ETL.	E
12	Žádost ODS o aktualizaci dat v DWH.	I
13	Žádost EIP o aktualizaci dat v ODS nebo jejich vyhledání.	I
14	Žádost EIP o aktualizaci dat v DWH nebo jejich vyhledání.	I
15	Žádost EIP o aktualizaci dat nebo jejich vyhledání prostřednictvím XQuery Gatewaye.	I
16	Žádost XQuery Gateway o aktualizaci dat v cílové aplikaci nebo jejich vyhledání.	E
17	Žádost aplikace o aktualizaci dat nebo jejich vyhledání prostřednictvím XQuery Gatewaye.	E
18	Žádost XQuery Gateway o aktualizaci dat v DWH nebo jejich vyhledání.	I
19	Žádost XQuery Gateway o aktualizaci dat v ODS nebo jejich vyhledání.	I
20	Žádost EIP o zprostředkování služby nebo procesu integračním serverem.	I

21	Žádost webového serveru o vygenerování stránky z EIP.	I
22	Žádost webového prohlížeče o vygenerování stránky z webového serveru.	E
23	Žádost ETL o aktualizaci dat v ODS nebo jejich vyhledání pro potřeby ETL.	I
24	Předání dávky dat z ODS ke zpracování v ETL.	I
25	Předání dávky dat z DWH ke zpracování v ETL.	I
26	Žádost ETL o aktualizaci dat v DWH nebo jejich vyhledání pro potřeby ETL.	I

tabulka 13 - Popis integračních cest v modelu (zdroj: autor)

Atomický popis integračních cest nám ovšem zakrývá, jak jsou cesty, resp. integrační komponenty, platformy kombinovány pro řešení integračních úloh. Integrační cestu či kombinaci více navazujících cest označíme za hlavní, jestliže v rámci integrační platformy zajišťuje plnění definované integrační úlohy. Integrační úlohy plněné integrační platformou jsem rozdělil do těchto skupin:

1. sestavení GUI pro uživatele,
2. sestavení stránky v EIP,
3. komunikace mezi aplikacemi,
4. aktualizace dat v ODS,
5. aktualizace dat v DWH

a pro tyto integrační úlohy jsem v tabulce 14 - Hlavní integrační cesty (zdroj: autor) popsal hlavní integrační cesty. Pro plnění integrační úlohy lze ve většině případů nalézt více možných kombinací. Tyto kombinace jsem rozdělil na optimální a alternativní, kde kritéria pro výběr optimální cesty jsou stanovena v souladu s integračními potřebami podniku:

1. Preferuje se použití standardizovaných postupů před nestandardními.
2. Preferuje se přístup k aktuálním datům před daty, která mohou být neaktuální (uložena mimo primární zdroj).
3. Preferuje se zprostředkovaná komunikace, která umožňuje volnou vazbu mezi komunikujícími aplikacemi (odstínění aplikací) a centralizovanou správu a dohled nad hlavní integrační cestou.
4. Preferuje se aplikace integračních komponent na integrační úlohy, na které jsou primárně navrženy.
5. Preferuje se univerzálnější řešení, které umožní další rozvoj a využití, před specializovaným a úzce zaměřeným.

Tato kritéria mohou být v reálném projektu potlačena jinými, zejména ekonomickými, což znamená, že optimální cesty nelze aplikovat jako dogma, ale v kombinaci s alternativními je vnímat jako koncepční vizi řešení.

Integr. úloha	**Ident. hlavní cesty**	**Vazba na integr. cesty**	**Popis**	**Optimální / alternativní**
Sestavení GUI pro uživatele	A	22+21	Stránka požadovaná webovým prohlížečem je předána přes webový server do EIP, který zajistí konsolidaci všech jejich částí - viz úloha Sestavení stránky v EIP - a předá výslednou stránku zpět přes webový server webovému prohlížeči.	optimální

	B	01	Webové UI aplikace (označme ji za primární) obsahuje vnořené obrazovky z jiných aplikací s webovým UI. Ve webovém prohlížeči dochází ke konsolidaci těchto obrazovek do GUI pro uživatele. Primární aplikace tak funguje částečně jako podnikový portál.	alternativní k A (menší centralizovaná správa a dohled nad integrační cestou)
	C	02	Obdoba cesty 01 bez využití webových technologií.	alternativní k A (menší centralizovaná správa a dohled nad integrační cestou)
	D	22+03	Stránka požadovaná z webového prohlížeče od webového serveru obsahuje odkazy na stránky generované UI aplikací. Webový server provádí konsolidaci těchto stránek před odesláním webovému prohlížeči. Webový server funguje jako primitivní podnikový portál.	alternativní k A (menší centralizovaná správa a dohled nad integrační cestou)
Sestavení stránky v EIP	E	04	Stránka požadovaná od EIP obsahuje odkazy na stránky generované UI aplikací. EIP si tyto stránky vyžádá, provede jejich konsolidaci.	optimální pro části stránky generované aplikacemi
	F	15	Požadovaná stránka pracuje s daty, která zpřístupňuje XQuery Gateway a jsou formátována pomocí EIP. EIP požádá XQuery Gatewaye o provedení operací s daty a výsledek zapracuje je do výsledné stránky.	optimální pro části stránky generované z dat
	G	20+05	Pro sestavení stránky je třeba provést služby poskytované integrovanou aplikací (např. registrace zákazníka). EIP požádá BPI o provedení této služby a podle výsledku služby sestaví stránku.	optimální pro stránky vyžadující provedení služby v integrované aplikaci.
	H	20+05	Požadovaná stránka pracuje s daty, která jsou dostupná v integrované aplikaci. EIP požádá BPI o provedení operací s daty ve zdrojové aplikace prostřednictvím služby.	alternativní k F (BPI není primárně určeno pro datovou integraci)
	I	20+07	Požadovaná stránka pracuje s daty, která jsou dostupná v ODS. EIP požádá BPI o provedení operací s daty (pouze čtení) v ODS prostřednictvím služby.	alternativní k F a L (ODS není primárním zdrojem dat)
	J	09	Pro sestavení stránky je třeba provést služby poskytované integrovanou aplikací (např. registrace zákazníka). EIP požádá o provedení této služby volanou aplikaci a podle výsledku služby sestaví stránku.	alternativní k G (těsná vazba na aplikaci)

	K	10	Požadovaná stránka pracuje s daty, která jsou dostupná ve zdrojové aplikaci. EIP požádá zdrojovou aplikaci o provedení operací s daty.	alternativní k F (těsná vazba na aplikaci)
	L	13	Požadovaná stránka pracuje s daty, která jsou dostupná v ODS. EIP požádá ODS o provedení operací s daty (pouze pro čtení).	alternativní k F a I (ODS není primárním zdrojem dat)
	M	14	Požadovaná stránka pracuje s daty, která jsou dostupná v DWH a nejsou v ODS. EIP požádá DWH o provedení operací s daty (pouze pro čtení).	alternativní k F (DWH není primárním zdrojem dat)
Komunikace mezi aplikacemi	N	06+05	Aplikace požádá BPI o provedení služby nebo procesu a integrační server požádá aplikace poskytující potřebné služby o jejich provedení. Případné výsledky zpracuje a předá zpět volající aplikaci. Existují dva možné přístupy podle toho, kde leží jádro podnikového procesu. Pokud je proces z velké části realizován volající aplikací, pak by BPI měl být volán pouze pro provedení požadovaných služeb a řízení procesu zůstává aplikaci. Naopak, pokud je jádrem procesu komunikace mezi více aplikacemi, pak by aplikace měly fungovat pouze jako spouštěče procesu (tedy vyvoláním procesu v BPI), který bude řízen integračním serverem. BPI lze použít jako poskytovatele služeb nebo procesů jako alternativní variantu, pokud aplikace nebo BPI není schopen naplnit požadavky vyplývající z optimálního přístupu užití cesty. Pokud např. aplikace, přestože jádro procesu leží na ní, není schopná podporovat procesní přístup, pak lze BPI užít jako manažera procesu.	optimální pro volání služeb mezi aplikacemi
	O	16	Pokud si aplikace potřebují vyměňovat pouze data bez volání služeb, mohou pro jejich předávání používat XQuery Gateway. Volající aplikace požádá XQuery Gateway o řízení operací s daty a ten je provede (např. operacemi v datech zdrojových aplikací).	optimální pro předávání dat mezi aplikacemi
	P	08+11	Pro výměnu dat mohou aplikace použít i předávání dat skrze ETL (může být rozšířeno i o data z ODS a DWH - cesty 24 a 25)	alternativní k O (méně univerzální než XQuery Gateway)

Aktualizace dat v ODS	Q	08+23	Data jsou ze zdrojových aplikací vylita do DSA a pomocí ETL transformována do struktur ODS. Alternativní způsoby nalití v reálném čase nelze použít, neboť by způsobily nadměrnou zátěž zdrojovým aplikacím.	optimální pro úvodní dávkové nalití dat a pro obnovení konzistence dat mezi ODS a aplikacemi - tzv. synchronizaci
	R	17+19	Změny v datech, které jsou synchronizovány do ODS, ve zdrojových aplikacích jsou detekovány a přeneseny přes XQuery Gateway do ODS.	optimální pro změnové aktualizace dat synchronizovaných do ODS
	S	06+07	Změny v datech, které jsou synchronizovány do ODS, ve zdrojových aplikacích jsou detekovány a přeneseny přes BPI do ODS.	optimální pro změnové aktualizace dat synchronizovaných do ODS, je-li změna potenciálním iniciátorem i jiných procesů než jen předání do ODS jinak alternativní k S (BPI není primárně určeno pro datovou integraci)
	T	20+05+07	Změny v datech, které jsou synchronizovány do ODS, pocházející z EIP (např. změna údajů o zákazníkovi) musí být přeneseny do zdrojové aplikace a následně mohou být přeneseny i do ODS, pokud není nastavena aktualizace dat přímo ze zdrojové aplikace. Transakčnost operace zajistí BPI.	alternativní k R, S a U (BPI zastupuje primární zdroj dat)
	U	15+16+19	Změny v datech, které jsou synchronizovány do ODS, pocházející z EIP (např. změna údajů o zákazníkovi) musí být přeneseny do zdrojové aplikace a následně mohou být přeneseny i do ODS, pokud není nastavena aktualizace dat přímo ze zdrojové aplikace. Transakčnost operace zajistí XQuery Gateway.	alternativní k R, S a U (XQuery Gateway zastupuje primární zdroj dat)

Aktualizace dat v DWH	V	08+26	Data jsou ze zdrojových aplikací vylita do DSA a pomocí ETL transformována do struktur DWH. Alternativní způsoby nalití v reálném čase nelze použít, neboť by způsobily nadměrnou zátěž zdrojovým aplikacím.	optimální pro úvodní dávkové nalití dat i pro postupnou dávkovou aktualizaci dat v DWH a pro obnovení konzistence dat mezi DWH a aplikacemi - tzv. synchronizaci
	W	12	Aktualizace dat mohou být předávány dávkově i z ODS namísto zdrojových systémů, pokud ODS obsahuje všechna potřebná data pro DWH.	alternativní k V (ODS není primárním zdrojem dat)

tabulka 14 - Hlavní integrační cesty (zdroj: autor)

V praxi se můžeme setkat s různými důvody pro výběr alternativní cesty. Identifikoval jsem tyto nejčastější příčiny:

- Chybí hlavní integrační komponenta pro optimální cestu.
- Z bezpečnostních důvodů není možné zvolit optimální cestu (např. data jsou dostupná pouze přes služby aplikace). Data jsou pro čtení zpravidla lépe přístupná než pro zápis a toho lze rovněž využít při hledání řešení integračního případu. Je například možné zpřístupnit data pro čtení rychlejší a možná méně systémovou cestou a pro méně užívaný zápis použít zcela jinou, bezpečnější cestu.
- Optimální cesta je v pro daný integrační případ nevhodná z důvodu výkonnosti - alternativní cesta je rychlejší za cenu méně systémového přístupu k řešení.
- Pro daný integrační případ nemá funkcionalita integrační komponenty žádný reálný ani potenciální přínos.

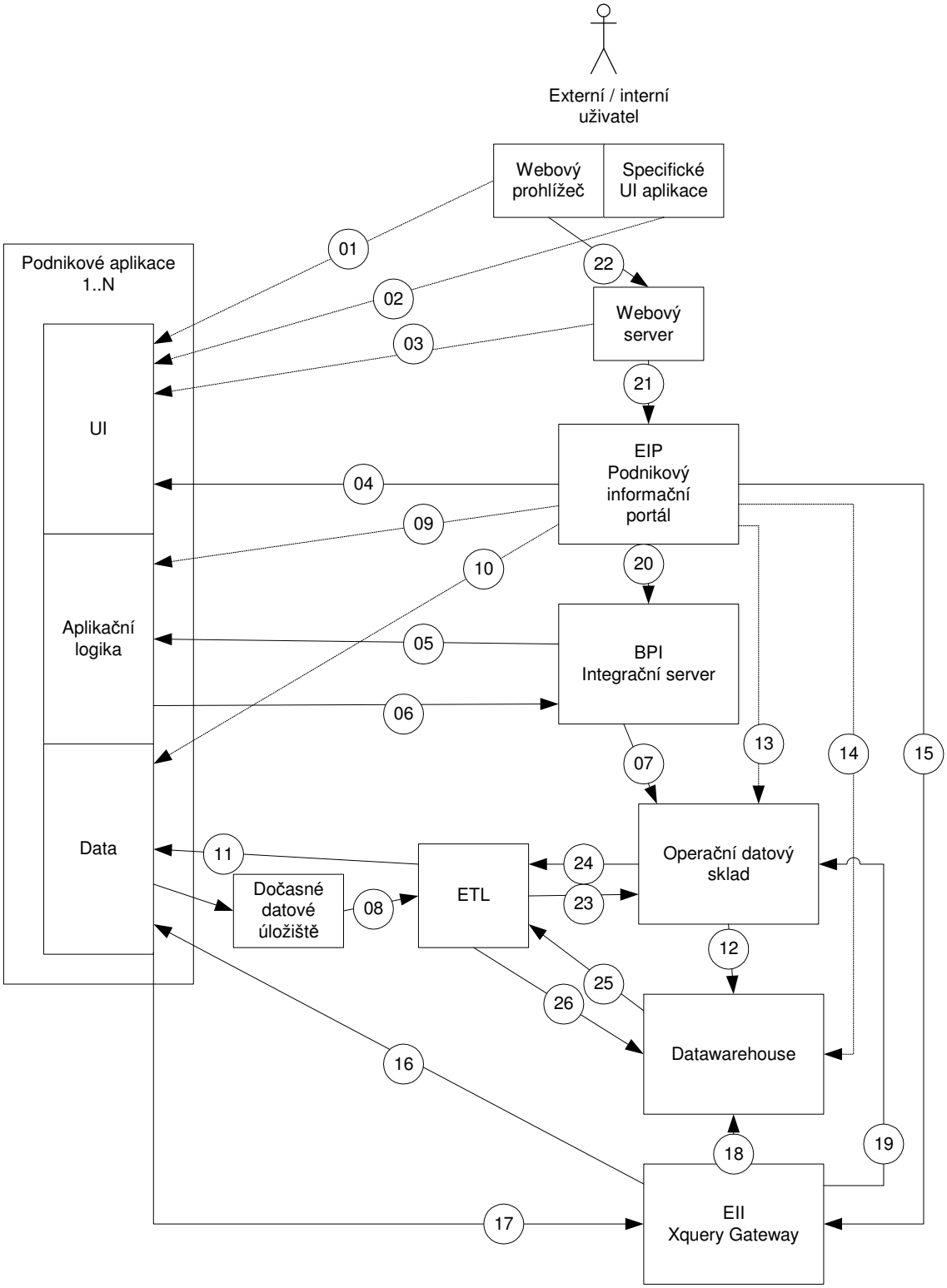
Co z modelu komplexní integrované architektury PIS lze odvodit?

1. Pro řešení integračních případů poskytují integrační komponenty principiálně několik odlišných postupů - integračních cest, a to především díky překrývání schopností těchto komponent. V důsledku se tento fakt využívá k **inkrementálnímu vylepšování** integračních schopností PIS přidáváním integračních komponent. Podnik si např. nejdříve vybuduje DWH, následně přidá ODS, poté BPI, pak EIP atd. a s každou komponentou přechází na optimálnější integrační cesty.
2. V některých případech jsou vedle optimální integrační cesty budovány a podporovány oficiálně i alternativní cesty, které slouží jako **nezávislá záloha** pro optimální integrační cestu. Typickým případem je způsob plnění ODS, kde jeho aktualizace v reálném čase přes BPI je zálohována dávkovým způsobem pomocí ETL.
3. Neexistuje komplexní integrační komponenta, ani platforma, která by optimálně řešila veškeré integrační potřeby PIS. Taková platforma by musela pokrývat všechny identifikované interní integrační cesty, neboli poskytovat schopnosti EIP, BPI, ETL, XQuery Gateway, webový server, ODS a DWH. Tyto jednotlivé **komponenty k sobě postupně konvergují**, tento trend je evidentní ze snahy dodavatelů těchto komponent rozšiřovat své portfolio produktů na chybějící komponenty. Příkladem této snahy je APS popsaný v kapitole 17.4.17 Aplikační platforma (APS - Application Platform Suite), který spojuje EIP a BIP.
   Bohužel se tato snaha projevuje pouze u jednotlivých dodavatelů, což směřuje k megalomanskému nástroji pro integraci dodávaném jedním dodavatel, což pro podnik patrně nebude akceptovatelné, neboť by se tak stal naprosto závislým na tomto

dodavateli. Jako efektivnější se jeví definování standardů pro rozhraní mezi integračními komponentami, což by umožnilo podniku vybrat si jednotlivé integrační komponenty podle svých potřeb. V současné době má podnik možnost si tyto komponenty vybrat rovněž jednotlivě, avšak jejich vzájemnou komunikaci musí řešit proprietárně jako komunikaci s jinými podnikovými aplikacemi.

4. Postupná optimalizace integračních cest (popsaná v bodě 1) vede k opětovnému řešení stejných integračních případů jinými komponentami, resp. postupy. Stejně tak budování alternativních cest (popsáno v bodě 2). Z bodu 3 vyčteme, že jednotlivé integrační komponenty nemají společné nástroje pro řešení integračních případů, neboť se chovají značně autonomně a nezávisle. Pokud se zamyslíme, čím je integrační případ definován (kapitola 9.2 Integrační proces), tak můžeme konstatovat, že z:
   - definice dat (metadat) ve zdrojových a cílových aplikacích, s kterými integrační případ pracuje,
   - definice služeb (abstrakce), které aplikace v PIS poskytují, a
   - business pravidel, které definují, jak se data transformují, které služby se kdy volají a jaká je návaznost kroků v integračním případu (procesu).

   Bude-li tedy existovat **společná a nezávislá repository pro metadata, definice služeb a definice business pravidel**, pak je možno s jejich využitím definovat nezávisle na integrační komponentě, či platformě, integrační případ. Jeho implementace pak bude záležitostí integrační platformy, která si potřebné konkrétní údaje o instancích dohledá v repository (více v kapitole 13.2 BRE v integračním projektu).
   Uveďme si příklad: Data o zákazníkovi v dané struktuře je třeba replikovat z aplikace X do ODS.
   Aby replikace začala, musí být splněny definované podmínky. Při replikaci se data popsaná metadaty aplikace X transformují na data popsaná metadaty ODS podle stanovených business pravidel (např. zkrácení adresy, doplnění business segmentu). Před zapsáním zákazníka do ODS musí být zákazník aktivován službou XY, kterou poskytuje aplikace Y. Definice metadat dat aplikace X a Y, business pravidel transformace dat i služby XY může být uložena v repository nezávisle na použitém integračním přístupu a technologii.



obrázek 29 - Model komplexní integrované architektury PIS (zdroj: autor)

12. Modely integračních nástrojů

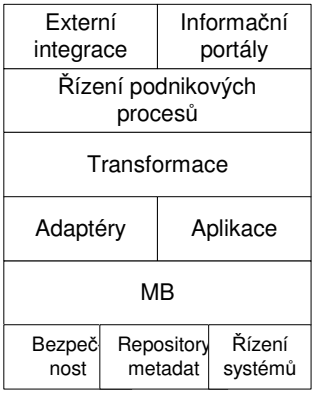
Jak jsme si ukázali v předešlé kapitole, integrační potřeby podniku naplňuje několik integračních nástrojů (integračních komponent PIS). V této kapitole si detailně popíšeme jejich funkcionalitu, resp. moduly, z kterých jsou složeny. Znalost těchto modelů nám umožní pochopit možnosti integračních nástrojů při řešení integračních úloh a zároveň je uplatníme při výběru vhodného integračního nástroje pro PIS. Budeme se detailně věnovat EIP, BPI (ESB), ODS, ETL a XQuery Gateway. DWH se nebudu explicitně věnovat, neboť jeho integrační schopnosti jsou principiálně podobné s ODS (zájemce odkazuji na [KIMBALL, 1998]). Modely představují plnou funkcionalitu zkoumaných integračních nástrojů.

12.1 Integrační server

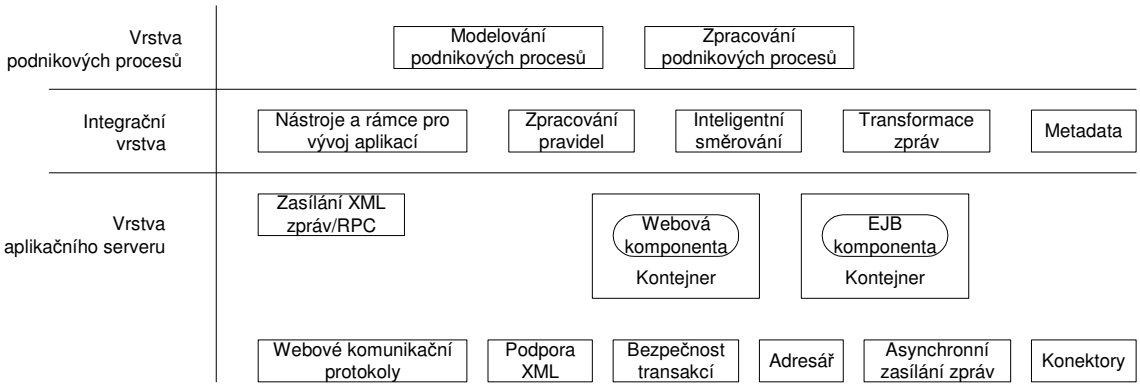
V informačních zdrojích (viz tabulka 15 - Přehled modelů integračních serverů v informačních zdrojích (zdroj: autor)) jsem nenalezl model, který by kompletně a přehledně pokrýval funkcionalitu současných integračních serverů. Vedle uvedených informačních zdrojů má vlastní model i každý dodavatel produktu na bázi integračního serveru (tyto jsou samozřejmě upravené pro marketingové potřeby dodavatele, proto je nelze uvažovat jako obecné modely).

Identifikace informačního zdroje	Pokrytá oblast
[HOHPE, 2002b]	viz obrázek 30 - Model integračního serveru (zdroj: [HOHPE, 2002b])
[SHARMA, 2001]	viz obrázek 31 - Model integračního serveru (zdroj: [SHARMA, 2001])
[LHEUREUX, 2003]	viz obrázek 32 - Model integračního serveru (zdroj: [LHEUREUX, 2003])
[PINKSTON, 2001]	viz obrázek 33 - Model integračního serveru (zdroj: [PINKSTON, 2001])
[BUSSLER, 2003]	viz obrázek 34 - Model B2B integračního serveru (zdroj: [BUSSLER, 2003])
[KUTÁČ, 2004]	Definuje tyto komponenty integračního nástroje - adaptéry, konektory, transformace dat, inteligentní směrování, správa metadat a úložiště dat, podpora webových služeb, řízení a modelování podnikových procesů, monitorování obchodních aktivit, správa a řízení, snadný a rychlý vývoj.

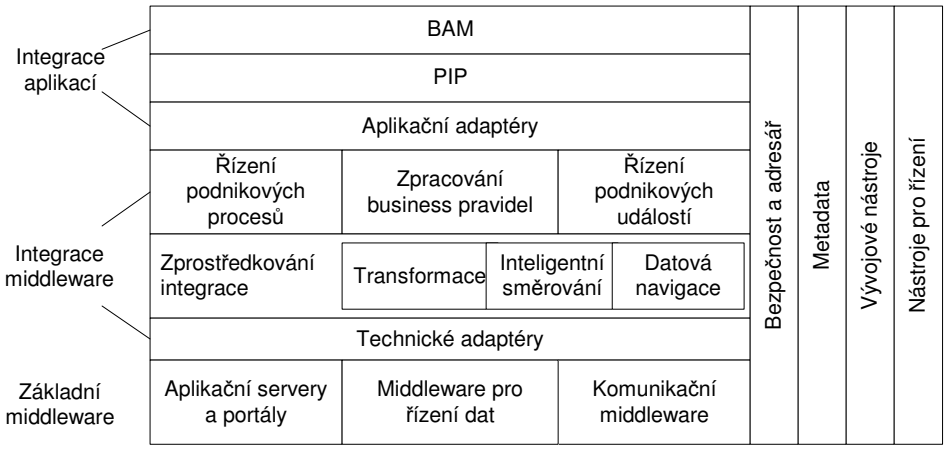
tabulka 15 - Přehled modelů integračních serverů v informačních zdrojích (zdroj: autor)



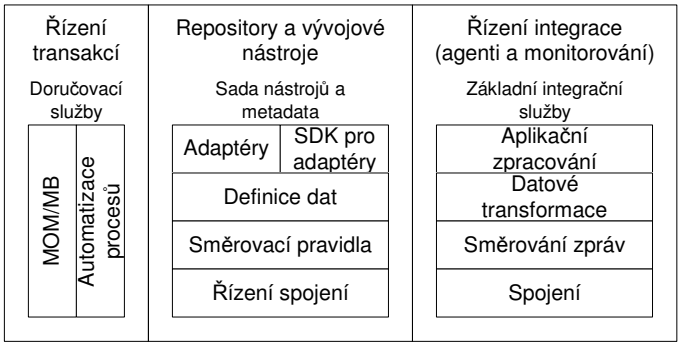
obrázek 30 - Model integračního serveru (zdroj: [HOHPE, 2002b])



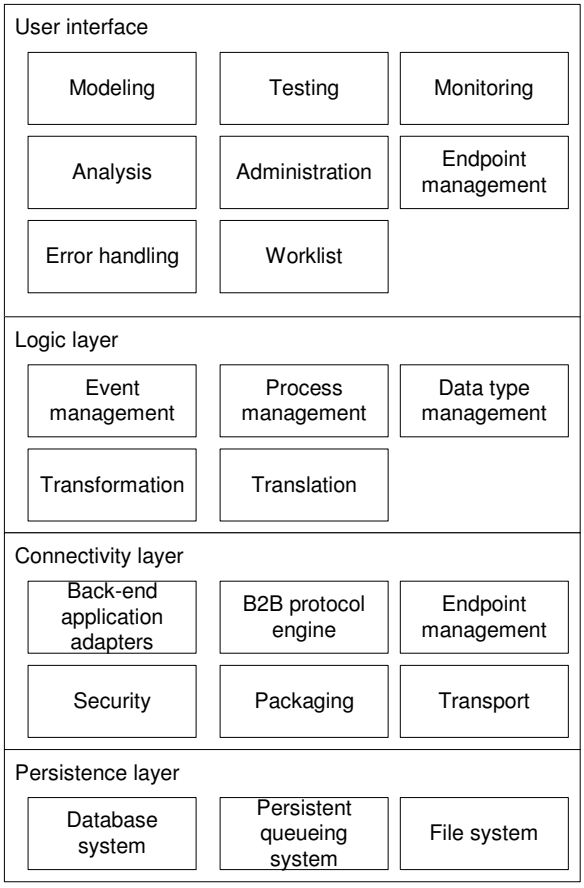
obrázek 31 - Model integračního serveru (zdroj: [SHARMA, 2001])



obrázek 32 - Model integračního serveru (zdroj: [LHEUREUX, 2003])



obrázek 33 - Model integračního serveru (zdroj: [PINKSTON, 2001])



obrázek 34 - Model B2B integračního serveru (zdroj: [BUSSLER, 2003])

Při sestavování vlastního modelu integračního serveru jsem čerpal z [CSC, 2001], [HOHPE, 2002b], [LHEUREUX, 2003], [SHARMA, 2001], [PINKSTON, 2001], [GOLDSTONE], [WOODGATE, 2004] a praktických poznatků.

Graficky je model integračního serveru zobrazen na obrázku 35 - Model integračního serveru (zdroj: autor) a obsahuje tyto moduly:

1. Runtime prostředí - provozní prostředí ohraničující komponenty pro rutinní provoz.
   a. Aplikační adaptéry - umožňují komunikaci mezi integrační platformou a integrovanými podnikovými aplikacemi.
      Při komunikaci s aplikací využívají komponent integračního serveru a zejména technologických adaptérů. Aplikační adaptéry se od technologických liší tím, že jsou sestaveny pro konkrétní aplikaci, její datové struktury a služby. Adaptéry překládají aplikačně specifickou komunikaci do událostí a detekují problémy při komunikaci s příslušnou aplikací.
      Jednotlivé adaptéry zpřístupňují služby integrovaných aplikací prostřednictvím SB. Příkladem těchto adaptérů jsou konektory pro hlavní ERP a CRM aplikace (SAP, Oracle, PeopleSoft, Siebel, Vantive).
   b. Technologické adaptéry - poskytují integrační technologie pro komunikaci s aplikacemi.
      Podporují jednotlivé standardizované komunikační protokoly, jako např. NetSQL, HTTP(s), FTP, SOAP a formáty zpráv jako např. XML a EDI. Příkladem těchto adaptérů jsou aplikačně nezávislé konektory (ODBC, JDBC, DB2, SQL, Text), middlewarové konektory (COM, CORBA, Web Services, J2EECA), mainframové konektory (IMSL, CICS), průmyslové formáty

(EDI/EDIFACT, X.12, SWIFT, HL7, RosettaNet) a konektory na integrační platformy třetích stran (MQSeries, TIBCO, MSMQ).

c. Zpracování podnikových procesů (Business Process Engine, též EE (Execution Engine [JANDOŠ, 2004]) - komponenta řídí běh procesů, které byly sestaveny komponentou pro modelování procesů.
   i. Message broker - zajišťuje předávání zpráv mezi aplikacemi.
   ii. Service broker - zajišťuje volání služeb mezi aplikacemi.
   K práci se službami se využívá tzv. registr služeb, který pro každou službu popisuje povinná a volitelná data, odpovědi a chybové zprávy. Tyto služby jsou využívány při modelování a běhu procesů.
   iii. Řízení událostí (Business Event Management) - zachytává a zpracovává události. Události mohou být vyvolány aplikací, zásahem uživatele nebo systémem.
   iv. Zpracování business pravidel (Business Rules Engine) - aplikuje definovaná business pravidla v rámci podnikových procesů. Business pravidlem může být např. postup při transformaci dat, podmínky pro rozhodování v procesu, posloupnost prováděných operací.
   v. Transakční monitor (Transaction Processing Monitor) - řídí provádění transakcí mezi aplikacemi.
   Většina podnikových procesů se skládá ze složených násobných transakcí, přičemž výsledek procesu závisí na tom, které z těchto transakcí se podařilo provést, případně zda se je podařilo provést všechny jako celek. Řízení transakcí se stará právě o tuto část.
   vi. Repository metadat - spravuje repository metadat a jejich užití. Metadata jsou data o podnikových procesech, událostech, datových objektech, parametrech napojení na aplikace, parametrech řízení systému, bezpečnostních informacích, pravidlech mapování dat, obchodních partnerech a další informace nutné pro běh a údržbu prostředí EAI. Tato data jsou uložena v repository a jsou využívána ostatními moduly integrační platformy.

d. Monitorování podnikových aktivit (Business Activity Monitoring) - umožňuje probíhající procesy monitorovat (tedy např. zjistit aktuální stav konkrétní objednávky) a vyhodnocovat pomocí nastavených metrik.

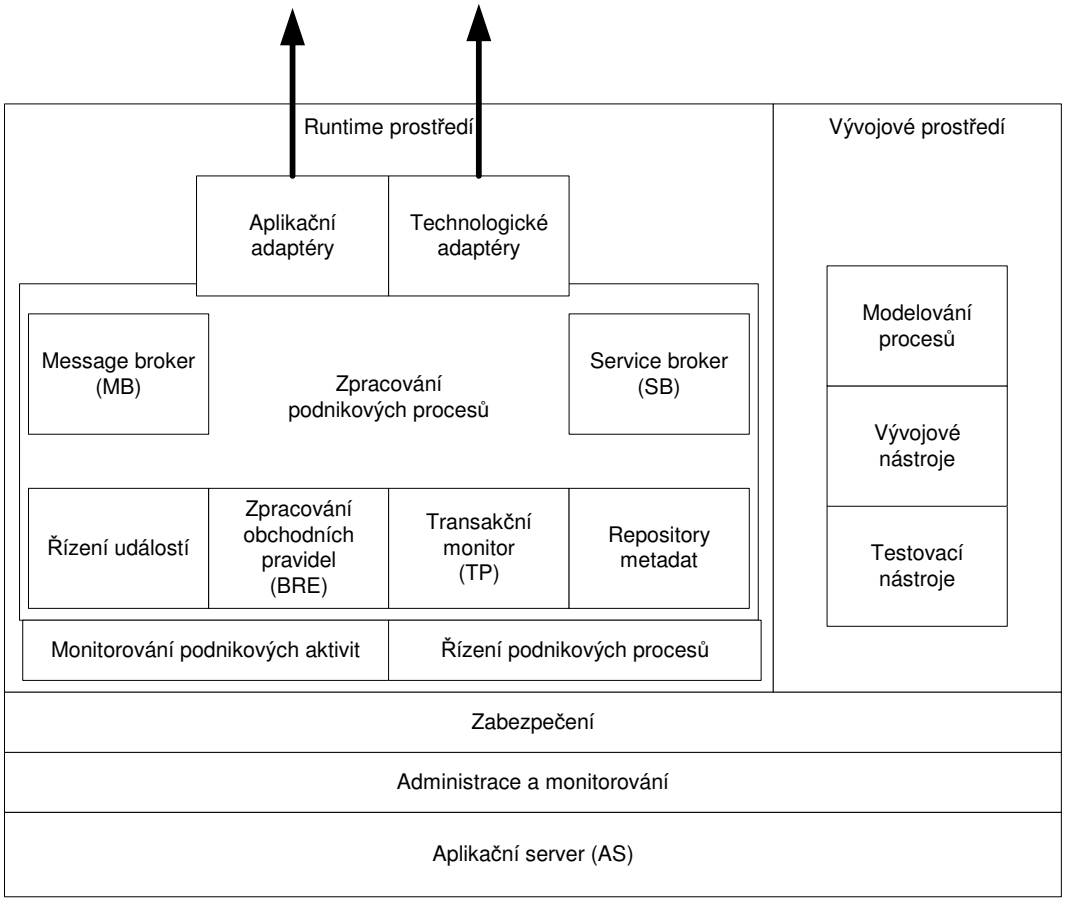
e. Řízení podnikových procesů (Business process management) - je určen pro uchovávání stavů běžících procesů a pro jejich případné změny uživatelem. V širším pojetí zahrnuje i nástroje pro modelování procesů a jejich monitorování ([PAPAZOGLOU, 2005], [NEWCOMER, 2004]). Detailní charakteristikou BPM a srovnáním produktů podporujících BPM se zabývá [MIERS, 2005]. [JANDOŠ, 2004] definuje BPM jako "činnost spočívající v průběžném (soustavném) provádění, monitorování, úpravě a inovaci podnikatelských procesů tak, aby odpovídaly podnikatelskému prostředí a podnikatelské strategii."

2. Vývojové prostředí - prostředí pro implementaci integračních případů.

a. Modelování procesů - umožňuje modelování podnikových procesů ve vazbě na služby poskytované integrační platformou a adaptéry aplikací.

b. Vývojové nástroje - umožňují úpravy a vývoj adaptérů a dalších komponent integrační platformy.
   Pomocí těchto nástrojů vývojáři připravují adaptéry k aplikacím, které je nemají předpřipravené. Rozhraní integrační platformy se liší dle dodavatele, adaptéry tedy nejsou přenositelné. V tomto směru je rozšířen standard J2EECA (J2EE Java Connector Architecture), který by měl vést k unifikaci adaptérů. Vývojovým jazykem adaptérů může být dle potřeby programovací jazyk C nebo Java, případně další.

   c. Testovací nástroje - umožňují testování vyvinutých adaptérů a procesů.
3. Sdílené komponenty pro obě prostředí:
   a. Zabezpečení - zajišťuje bezpečnost komunikace a chodu integrační platformy. Řeší úlohy týkající se autentizace a autorizace uživatelů a aplikací, šifrování a zajištění integrity přenášených dat, session managementu a auditování prováděných činností.
   b. Administrace a monitorování - jelikož se jedná o systém "mnoha pohyblivých částí", vyvstává nutně potřeba celý tento systém monitorovat a řídit. Administrátor může spouštět a zastavovat jednotlivé komponenty systému, monitorovat výkon, aktualizovat komponenty (tj. adaptéry, metadata, business pravidla, procesy), opravovat nebo restartovat chybné transakce.
   c. Aplikační server - poskytuje technologie pro chod integrační platformy a správu komponent integrační platformy.
   Jednotlivé komponenty integrační platformy vycházejí z jednotného technologického prostředí (např. J2EE, .NET), které má v sobě implementované sdílené technologické standardy. Komponenty musí být rovněž administrovány a monitorovány.



obrázek 35 - Model integračního serveru (zdroj: autor)

K vybraným komponentám jsou dále uvedeny upřesňující informace.

12.1.1 Aplikační adaptéry

Aplikační adaptéry jsou klíčem pro aplikační integraci, proto zde pro úplnost popíšeme způsob fungování adaptérů. Komunikace mezi aplikacemi probíhá jako sled několika navzájem na sebe navazujících kroků :

1. Generování požadavek na službu - aplikace zjistí potřebu pro svou další činnost získat data od jiné aplikace nebo provést činnost, kterou má na starosti jiná aplikace. V takovém případě zformuluje požadavek na službu včetně potřebných vstupních parametrů a odešle jej na ke zpracování na integrační platformu.
2. Předání požadavku - integrační platforma požadavek analyzuje, prostřednictvím metadat určí aplikaci, které je požadavek určen. Dále aplikuje své vlastní řídící mechanismy, jako je např. řízení transakcí, bezpečnost. Zároveň aplikuje funkce pro transformace dat, čímž požadavek resp. zprávu resp. data připraví pro cílovou aplikaci.
   Poté pomocí modulu message broker a adaptéru předá požadavek této aplikaci ke zpracování. Adaptér v této fázi rovněž kontroluje, zda požadovaná služba odpovídá svým názvem a parametry službě, kterou volaná funkce zpřístupnila prostřednictvím adaptéru ostatním, resp. dané konkrétní aplikaci.
3. Zpracování požadavku - cílová aplikace provede požadovanou službu, což může být např. inicializace jiné akce, zapsání či vyhledání dat v databázi. Výsledek poté předá zpět integrační platformě (toto zajišťuje adaptér).
4. Předání výsledku - integrační platforma (obdobně jako v bodě 2) nasměruje výsledek volající aplikaci.
5. Zpracování výsledku - původně volající aplikace přijme výsledek svého požadavku na službu prakticky stejně jako by realizovala volání své interní procedury resp. funkce a může s výsledkem dále pracovat. Tato klíčová vlastnost integrace se obecně nazývá transparentností fyzického umístění procesů klienta (volané aplikace) a serveru (volající aplikace) ([DOHNAL, 1997], str. 190).

   Poznámka: Kroky 4 a 5 nejsou povinné, pokud nejsou aplikacemi vyžadovány.

Adaptéry můžeme rozlišovat podle poskytované funkcionality [IBM]:

- Pouze konektivita (např. CICS Transaction Gateway, SAP R/3 RFC).
- Konektivita a abstrakce rozhraní (např. VisualAge for Java Record Libraries, Lotus DECS, SAP Connector) (též tenký adaptér dle [LINTHICUM, 2002]).
- Konektivita, abstrakce rozhraní a business logika (např. VisualAge for Java Enterprise Access Builders, Access Builder for SAP R/3) (též tlustý adaptér dle [LINTHICUM, 2002]). Tlusté adaptéry poskytují výrazně širší využití a přidanou hodnotu pro podnik, jejich vývoj je však náročnější (dle [LINTHICUM, 2002] až šestinásobně).

Podle umístění adaptéru rozlišujeme (dle [LINTHICUM, 2002]) aplikační adaptéry:

- centralizované - běží na integrační platformě, jedná se o tenké adaptéry pouze mapující API integrační platformy na API aplikace
- distribuované - běží na integrační platfomě a v aplikaci, jedná se o tlusté adaptéry s přidanou business logikou

Podle poskytovaného výstupu rozdělujeme adaptéry na poskytující (dle [LINTHICUM, 2002]):

- statická data,
- dynamická data (na základě parametrů),
- výsledná data funkce,
- vzdálenou službu,
- abstrakci (vzdálenou službu, která může být modifikována dle potřeb volající aplikace).

12.1.2 Zpracování podnikových procesů

Existují různé přístupy jak přistoupit ke zpracování podnikových procesů integračním serverem - [IBM] uvádí:

1. Rozšířit dosah podnikového procesu integrací s dalšími aplikacemi.
2. Spojení dvou oddělených podnikových procesů do jednoho společného procesu.

3. Oddělení podnikového procesu od aplikační logiky jeho implementací procesu do integračního serveru. Tím se řízení procesu dostane kompletně pod kontrolu tohoto modulu.

[PALMER, 2005] poskytuje přehled o aktuálním stavu BPM, včetně srovnání BPM nástrojů.

Můžeme se setkat i s předpřipravenou podporou podnikových procesů pro konkrétní integrační platformu. Tyto předpřipravené procesy se označují zkratkou **PIP** (Packaged integrating processes). Příkladem může být řešení **UAN** (Universal Application Network) poskytovaný společností Siebel Systems, což je integrační řešení obsahující integrační platformu některého z partnerů Siebelu a předpřipravené podnikové procesy, objekty (Common Objects) a transformační pravidla (Transformations) (viz [JONES, 2003]).

Rovněž další dodavatelé velkých podnikových aplikací nabízejí jako součást řešení vlastní integrační platformu s předpřipravenými procesy, např. Oracle, J.D.Edwards, SAP (pro interní integraci Exchange Infrastructure a pro externí Business Connector postavený na technologii WebMethods). Podle [LHEUREUX, 2003] se nedoporučuje tyto integrační platformy využívat, neboť nejsou dostatečně univerzální a flexibilní.

12.1.3 Message broker

Tento modul poskytuje následující služby:

1. Zasílání zpráv mezi zdrojovou a cílovou aplikací.
2. Adresování a routování (inteligentní směrování s kontrolními body) - volající aplikace nemusí znát přesné umístění volané aplikace, dokonce ani identifikaci volané aplikace. Správné nasměrování požadavku zajistí MB na základě definovaných indicií od volané aplikace. Toto mimo jiné umožňuje v případě přetížení aplikace přesměrovat na jinou aplikaci poskytující stejnou službu.
3. Zabezpečení přenášených dat.
4. Komunikace s aplikačními adaptéry - volající aplikace nemusí znát syntaxi ani sémantiku volané aplikace.
5. Řešení synchronizace vstupů a výstupů (časová transformace).

Message broker pracuje s několika seznamy údajů [LUBLINSKY, 2001]:

- událostmi a zprávami (zprávy jsou výsledkem událostí),
- odesílateli zpráv (producers),
- příjemci zpráv (consumers) a
- obsahem zpráv (za účelem identifikace společných entit zpráv).

Integrační server potřebuje dočasně ukládat různá data, např. zprávy, kdy se čeká na potvrzení přijetí zprávy aplikací, či transakci, nebo pro uchovávání historie zpráv. S touto komponentou se můžeme setkat i samostatně pod názvem **Message Store**).

12.1.4 Zpracování business pravidel

Součástí této komponenty je i transformace dat, která provádí následující služby:

1. On-line konverze mezi specifickými i standardními formáty dat.
2. Agregace a rozdělování datových výstupů dle možných scénářů (viz také [IBM] a [OVUM, 2001]):
    a. one-to-one (mapování mezi dvěma aplikacemi),
    b. one-to-many (vytváří se rozdílné výstupy z jednoho vstupu),
    c. many-to-one (z mnoha zdrojů se vytváří komplexní výstup),
    d. many-to-many (z mnoha zdrojů se vytváří rozdílné výstupy dle potřeb aplikací).

3. Mapování hodnot (viz také [JEAN-CHARLES, 2001] a ([HOHPE], kap. Message Transformation Patterns)) - výsledná hodnota vzniká:
    a. kopírováním vstupní hodnoty,
    b. nastavením defaultní hodnoty,
    c. dohledáním hodnoty k identifikátoru,
    d. sloučením, nebo rozdělením hodnot,
    e. změnou pořadí částí hodnoty (zprávy), nebo
    f. kombinací výše uvedeného.

Podle způsobu definice transformace dat rozlišujeme tyto přístupy [JEAN-CHARLES, 2001]:

- Vytvořený na zakázku - postup transformace je uložen přímo v programovém kódu.
- Generický - postup transformace se řídí podle metadat, resp. pravidel, což je sice pomalejší, avšak umožňuje snadnou změnu.
- Hybridní - kombinace výše uvedených.

12.1.5 Transakční monitor

V ([DOHNAL, 1997], str. 208) nalezneme definici transakčního zpracování - je charakterizováno náhodným výskytem menších úloh (transakcí), které musí být zpracovány s relativně krátkou dobou odezvy. Podstatné je, že transakce není možné předem plánovat, přicházejí náhodně a mohou se překrývat. Je rovněž důležité si uvědomit hierarchickou logickou strukturu celého procesu, která je následující (viz také [DOHNAL], str. 192):

Proces - subproces (= příkaz) - operace - transakce - funkce.

Transakce ([DOHNAL], str. 208) je definována jako soubor aktivit, definovaný jako celek čtyřmi vlastnostmi - nedělitelnost, konzistence, izolovanost, stálost.

Úlohou transakčního monitoru se zabývá též [SESSIONS, 2003], nazývá jej DTC (Distributed Transaction Coordinator), resp. LCTM (Loosely Coupled Transaction Manager), který umožňuje rozdělení transakce na menší transakce (schované ve službách), které dostávají informace i o celé transakci.

12.1.6 Repository metadat

Způsob ukládání metadat do repository není jednotný, neexistuje dohodnutý standard. Podle [MORAGENTHAL, 2005b] se v současné době používají tři odlišné technologie:

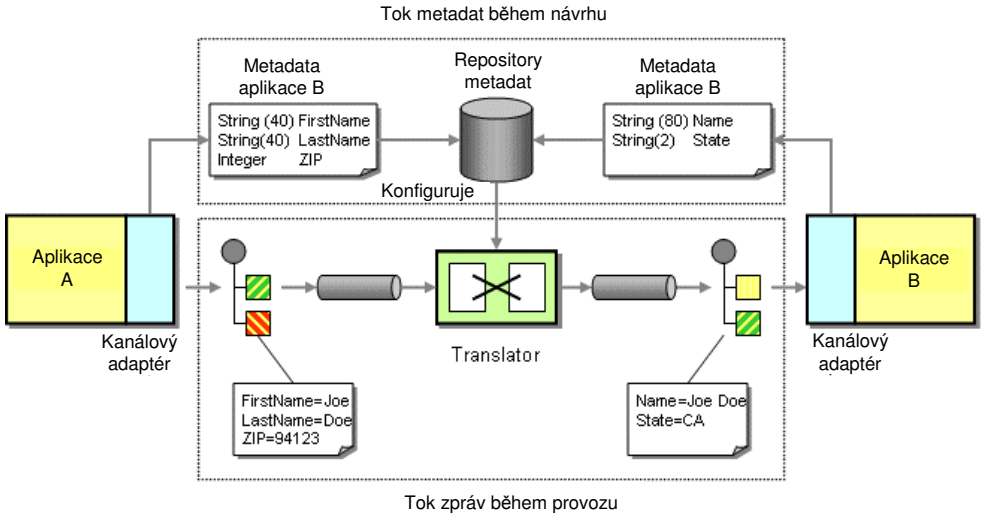
1. Relační databáze (RDMS - Relation Database Management System).
2. Objektová databáze, která proti relační databázi podporuje hierarchické uspořádání metadat, avšak nepodporuje standard SQL.
3. Adresářové služby (Directory Services), které se dále rozdělují na:
    a. LDAP (Lightweight Directory Access Protocol - ([MORAGENTHAL, 2005b] poznamenává, že dosud není jasné, zda LDAP je dostatečně robustní pro využití jako repository podnikových metadat).
    b. NDS (NetWare Directory Services).
    c. ADSI (Active Directory Services Interface).

Standardizací v této oblasti se zabývalo uskupení firem Meta Data Coalition, které definovalo **OIMMDIS** (Open Information Model a Meta Data Interchange Specification) ([MDC, 1999] a [MDC, 2000]) určený pro přenášení metadat. Tento model je technologicky nezávislou specifikací klíčových typů metadat, s kterými se můžeme setkat v provozním prostředí, datovém skladu a znalostním managementu a může být implementován prostřednictvím UML, XML a SQL. Skládá se z následujících částí:

1. modely analýzy a návrhu,
2. objektové a komponentové modely,

3. modely databází a datových skladů,
4. modely znalostního managementu,
5. podnikové modely.

Vlastní metadata se využívají např. při transformaci zpráv, jak ukazuje obrázek 36 - Příklad využití repository metadat (zdroj: [HOHPE]).



obrázek 36 - Příklad využití repository metadat (zdroj: [HOHPE])

12.1.7 Monitorování podnikových aktivit

Komponenta monitorování podnikových aktivit (BAM) je nepovinná a umožňuje :

- zachytávat události v procesech,
- vyhodnocovat zachycené události pomocí filtrů a business pravidel,
- upozorňovat na vybrané události.

BAM definuje společnost Gartner jako "koncept poskytování přístupu ke kritickým podnikově-výkonnostním indikátorům v reálném čase pro zlepšení rychlosti a efektivity podnikových operací".

Praktická ukázka v prostředí ORACLE je prezentována v [MATĚJŮ, 2005].

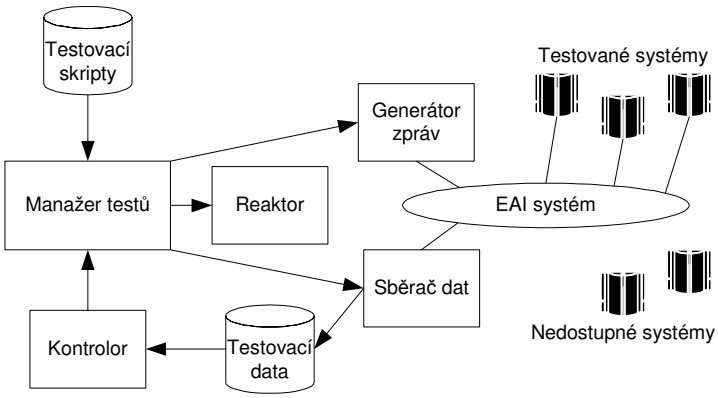
12.1.8 Řízení podnikových procesů

Podle [NEWCOMER, 2004] se komponenta BPM skládá se 3 částí - **modelování, spouštění a monitorování procesů**. Na tyto části je přímo připojena komponenta BAM a společně tak tvoří kompaktní celek. Funkcionalita jednotlivých částí je dle [NEWCOMER, 2004] charakterizována takto:

- Procesní modelování - grafický modelovací nástroj pro navrhování procesů, technické modelování toků, modelování a monitorování metrik, navrhování spolupráce, simulaci a animaci podnikových procesů, dokumentaci procesů, flexibilní navrhování formulářů, generování reportů, administraci uživatelů a bezpečnosti.
- Spouštění procesů - nástroj pro zpracovaní procesů, zpracování business pravidel (BRE, viz samostatná komponenta), řízení eskalací a vyjímek, řízení zpracování, delegaci úloh, směrování (ad hoc, podle uživatele, pravidla, role), plánování, kompenzační transakce, obnovu procesu, spolupráci mezi uživateli.
- Monitorování procesů - nástroj pro přehled běžících i ukončených procesů, sledování stavu procesu, pozastavování procesu, změnu priorit procesů, restartování procesu.

12.1.9 Testovací nástroje

Testovací nástroj integračního serveru by měl podporovat nejrůznější protokoly pro zasílání zpráv, XML standardy a rozličné způsoby formátování obsahu. Cílem testování je ověřit chování integrační platformy a na ní napojených aplikací ve všech klíčových situacích. Testování probíhá dle testovacích scénářů (test cases) s pomocí testovacích skriptů pro automatizované testování. Obrázek 37 - Testovací schéma na integrační platformě (zdroj: [POOL, 2003]) popisuje způsob komunikace mezi testovacím nástrojem a výkonnou částí integrační platformy (viz také [HOHPE]).

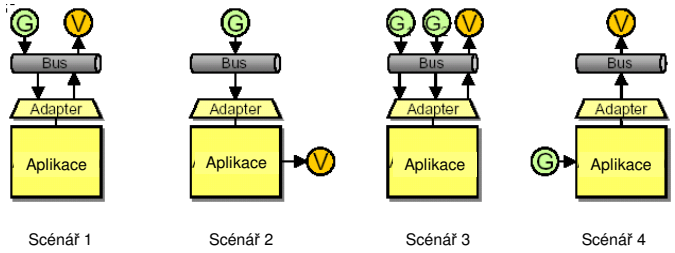


obrázek 37 - Testovací schéma na integrační platformě (zdroj: [POOL, 2003])

Testovací nástroj se skládá s těchto komponent:

1. Manažer testů zajišťuje spouštění, zaznamenávání průběhu a vyhodnocování testů.
2. Generátor zpráv pro vpouštění dat do integrační platformy.
3. Reaktor simulující požadavky a nebo odpovědi během transakcí (zastupuje komponentu systému, která není přímo testována).
4. Sběrač dat sbírající testovací data.
5. Kontrolor vyhodnocující, zda chování systému odpovídá požadavkům.

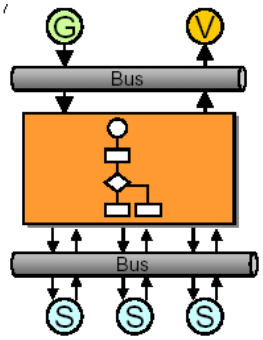
Tento nástroj by nám měl umožnit testovat i jednotlivé části integračních případů (zaměřené na funkčnost adaptérů), jak dokumentuje obrázek 38 - Testovací scénáře pro test adaptérů (převzato z (zdroj: [HOHPE, 2002b]).



obrázek 38 - Testovací scénáře pro test adaptérů (převzato z (zdroj: [HOHPE, 2002b])

Poznámka k obrázku: G - Generátor, V - Kontrolor, Bus - integrační platforma.

Ze stejného informačního zdroje se dozvíme i základní testovací scénář integračního procesu. Testování procesů je náročnější vzhledem k jejich komplikovanosti a tím i velkému počtu možných stavů - viz obrázek 39 - Testovací scénář pro test integračního procesu (zdroj: [HOHPE, 2002b]).



obrázek 39 - Testovací scénář pro test integračního procesu (zdroj: [HOHPE, 2002b])

Poznámka k obrázku: G - Generátor, V - Kontrolor, S - Reaktor, Bus - integrační platforma.

Výsledkem testování je konstatování, zde test (přesněji testovací příklad) proběhl bez závad nebo s chybami (a ty se podrobně popíší). Následuje oprava chyb a nové testování.

12.1.10 Zabezpečení

Základním pravidlem bezpečnosti informačních systémů je skutečnost, že nejslabší článek řetězu určuje pevnost celého řetězu. Nelze tedy integrovat aplikace, aniž bychom umožnili nekontrolovatelný přístup k podnikovým aplikacím, neboť tím by se právě tento prvek stal pravděpodobně oním nejslabším článkem.

V souvislosti s integrací aplikací často dochází k přesnějšímu definování odpovědnosti vlastníků jednotlivých aplikací a dat za jejich zabezpečení. Je nutné definovat hranici, kdy je za prováděnou službu odpovědný vlastník aplikace a kdy správce integrační platformy. To není vždy tak jednoduchá otázka, jak by se mohlo zdát, když si připomeneme problematiku provádění transakcí.

Z implementace EAI do podniku vyplývá tedy poměrně důležitý závěr. Nevhodně navržené řešení zabezpečení funkcí integrační platformy ohrozí všechny integrované aplikace podniku včetně jejich dat, naopak řešení provedené s přihlédnutím ke všem podstatným rysům integrace způsobí posílení zabezpečení aplikací přesnějším definováním částí bezpečnostní politiky a umožní bezpečnost na úrovni EAI řídit. Řízení bezpečnosti na úrovni EAI nahradí často neprůhledně definované a nechráněné komunikační kanály realizované dosud přímo mezi jednotlivými aplikacemi jako dočasné provizorní řešení.

Koncept EAI s sebou přináší konkrétní bezpečnostní prvky, které umožňují jednoduchou implementací zvýšit bezpečnost integrovaného informačního systému podniku a v aplikaci do oblasti e-businessu jsou jedním z předpokladům pro otevření podniku směrem k intranetu či extranetu. Za jejich korektní implementaci je zodpovědný člověk v roli specialisty na informační bezpečnost, který se zároveň podílí i na analýze a návrhu jednotlivých služeb a adaptérů.

Oblast systému	Oblast bezpečnosti	Zranitelné místo
Integrované podnikové aplikace	Důvěrnost	Zdrojová data mohou být přístupná z integrační platformy.
		Uživatelské heslo integrační platformy může být prozrazeno.
		Rozhraní používané integrační platformou může obsahovat bezpečnostní díru, která se neprojeví u standardního rozhraní (GUI).
		GUI pro konfiguraci bezpečnostních profilů uživatelů je obcházeno.

		EAI uživatelé/procesy mohou získat přístup k aplikaci, přestože nejsou aplikací authorizováni.
	Integrita	vše výše uvedené
		Příchozí data mohou být poškozena.
	Dostupnost	DoS (Denial of Service) útok proti aplikaci může být proveden z integrační platformy.
		Chod aplikace může být ovlivněn, pokud není integrační platforma dostupná.
Integrační platforma	Důvěrnost	Data řízená integrační platformou ke sdílení mezi aplikacemi mohou být přístupná neautorizované straně.
		Data mohou být odposlechnuta během komunikace.
		Neautorizovaná strana může získat přístup k dočasným datům integrační platformy a získat tak citlivé informace.
	Integrita	Data procházející integrační platformou mohou být narušena.
		Integrační platforma může generovat falešná data a zasílat je aplikacím.
	Dostupnost	DoS (Denial of Service) útok proti integrační platformě může být proveden z integrované aplikace.
	Provozuschopnost	Procesní tok může být narušen obejitím legitimních business operací.
Změnový proces		Mnoho-komponentovou, více systémovou strukturu je obtížné kontrolovat a zajistit konzistenci software během vývoje, testování a provozu.

tabulka 16 - Přehled zranitelných míst uvnitř integrovaného systému (zdroj: [VARLAMOV, 2002])

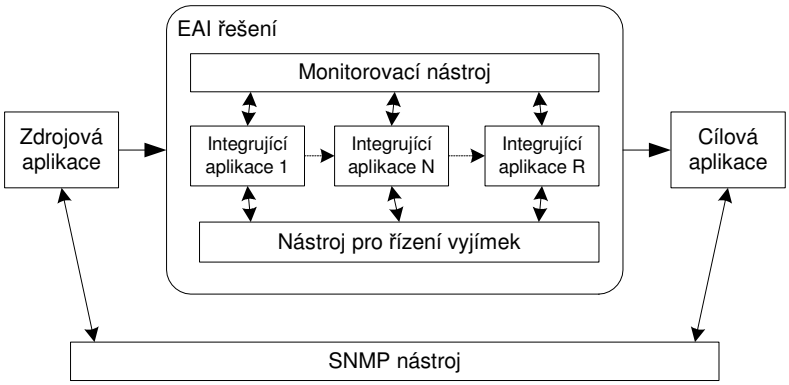
Lze identifikovat následující bezpečnostní prvky integrační platformy (touto problematikou se zabývá např. [JAISWAL]):

1. Autentizace systémových uživatelů a aplikací zasílajících nebo přijímajících zprávy integrační platformy - na úrovni integrační platformy je možno a nutno definovat, které služby volané aplikace budou k dispozici té které volající aplikaci.
2. Přidělování přístupových práv k aplikacím a jejím datům - definováním služeb jsou přesně vymezena práva volající aplikace k funkcím a datům volané aplikace. V žádném případě se nedoporučuje umožnit tzv. plný přístup k datům volané aplikace např. tím, že uvolníme k použití službu provádějící jakýkoli SQL (INSERT, UPDATE, DELETE, SELECT) příkaz.
   Ani povolení nespecifikovaného nedestruktivního příkazu SELECT nelze v žádném případě doporučit, neboť případný útočník by prostřednictvím tohoto příkazu mohl daný software či hardware zahltit a způsobit zborcení celé aplikace.
3. Řízení transakcí - tento prvek byl již detailněji zkoumán v jedné z minulých kapitol. Pomocí prostředků middleware lze definovat jednotlivé stavy, které je nutno z pohledu transakcí ošetřit a s vazbou na procesní integrační nástroj definovat i jednotlivé scénáře jejich řešení.
4. Přidělování práv (např. přijímání zpráv) jednotlivým uživatelům nebo skupinám uživatelů - v kombinaci s konceptem a nástroji SSO (Single Sign On) (např. LDAP), jehož hlavní myšlenkou je automatické získání přístupu k množině definovaných aplikací po úspěšném přihlášení k jedné aplikaci z této množiny, lze definovat práva uživatelů ke skupině aplikací i k aplikacím jednotlivě. V roli uživatele může samozřejmě vystupovat i jiná aplikace identifikovaná počítačovým uzlem.

5. Šifrování přenášených zpráv - přenášené zprávy, neboli data je možno zabezpečit šifrováním, nejčastěji pomocí technologie SSL (Secure Socket Layer) používající technologii PKI. Integrační platforma často používá své specifické protokoly pro přenos, které není jednoduché dešifrovat ani samotné. Pokud chceme přenášená data šifrovat, stojíme před dvěma problémy. Prvním z nich je zvýšení časového intervalu předání zprávy dané aplikaci, což je způsobenou časem nutným pro šifrování a dešifrování obsahu zprávy. Druhým problémem je podmínka, že daná šifrovací technologie musí být podporována na obou stranách, tedy jak na straně middleware, tak na straně aplikace. Málokterá vnitropodniková aplikace má připravený modul s nejnovější technologií SSL včetně vystaveného certifikátu. V některých případech lze tento problém řešit přidáním předpřipraveného modulu dodávaného s integrační platformou do stávající aplikace v rámci jejího přizpůsobování vazbě na integrační platformu.
6. Neporušenost zprávy - integrační platformou používané protokoly a kontrolní mechanismy jsou schopny zajistit neporušenost předávané zprávy.
7. Dynamické konfigurování a uzavírání otevřených spojení (session) mezi obchodními partnery v případě, kdy jsou spojení otevírána na delší dobu.
8. Auditní protokoly - integrační platforma je schopna iniciovat a vytvářet záznamy o prováděných službách, resp. předávaných zprávách, tzv. auditní protokoly, které mohou sloužit jako důkazní materiál v případě sporu i materiál pro analýzy.
9. Nástroj pro IDS (Intrusion Detection System) - integrační platforma standardně poskytuje možnost připojení speciálního nástroje nazývaného IDS, který slouží pro detekci útočníků v síti. V realitě je integrační platforma dalším softwarovým prvkem, který je navíc do značné míry specifický, který musí útočník oklamat nebo ovládnout, aby se mohl svobodně pohybovat ve vnitropodnikovém informačním systému.
10. Specifikace funkcí a jejich parametrů - pokud jsou služby integrační platformy, potažmo aplikací navrženy vhodně, má útočník možnost využít pouze operace striktně definované službami, ke kterým navíc nemá žádnou dokumentaci a nezná jejich názvy ani parametry. Navíc pomocí sledování auditních protokolů lze snadno zjistit jeho neplatné pokusy, neboť všechny zainteresované strany (tedy volající i volané aplikace znají syntaxi přesně) a nezpůsobují chyby.

12.1.11 Administrace a monitorování

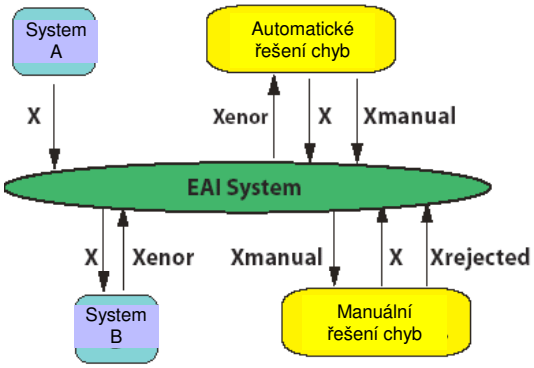
Nástroj pro monitorování slouží pro sledování aktuálního stavu systému (integrační platformy i aplikací) i pro zjišťování stavu systému v minulosti. Sledují se zejména problémové stavy (např. zahlcení, porušení bezpečnosti) a statistické údaje (např. vytížení systému) a zaznamenávají se podstatné údaje, sloužící pro zpětné dohledání nastání chybových stavů. Obrázek 40 - Monitorování v EAI (zdroj: [TOTLANI, 2004]) znázorňuje umístění monitorovacího nástroje v systému.



obrázek 40 - Monitorování v EAI (zdroj: [TOTLANI, 2004])

Administrace integrační platformy spočívá hlavně v řešení problémů integrační platformy, správě modulů platformy a realizaci upgradů platformy (včetně adaptérů).

Obrázek 41 - Detekce chyb v integrovaném PIS (zdroj: [POOL, 2003]) popisuje standardní způsob řešení chybových zpráv v systému. Systém A generuje zprávu pro systém B, který ji však odmítne převzít a označí ji chybovým příznakem (razítkem). Takto označené zprávy jsou zpracovávány modulem pro automatické zpracování chybových zpráv (Automated Fallout Management), který se zprávu pokusí opravit a předat systému B. Pokud neuspěje, předá zprávu k manuální opravě (Manual Fallout Management).



obrázek 41 - Detekce chyb v integrovaném PIS (zdroj: [POOL, 2003])

Chybové zprávy (též vyjímky) je možné rozdělit do těchto skupin [TOTLANI, 2004]:

1. Systémové - mají vliv na fungování celého systému, typicky se může jednat např. o ztrátu napojení na databázi, chyby na úrovni socketů, nepřístupné soubory, nedostatek paměti, výpadky sítě.
2. Datové - vznikají v případě chybných datových typů, porušení datových vazeb, chybného mapování, apod.
3. Procesní - nastávají v případě porušení business pravidel.
4. Bezpečnostní - jedná se např. o neautorizovaný přístup, vypršené certifikáty, problémy při autentikaci.

12.2 ODS

ODS je standardní databáze, do které jsou ukládána aktuální data pro okamžité využití připojenými aplikacemi. Zpravidla se jedná o údaje o zákaznících, produktech, produktových instancích (tedy jaké produkt má zákazník nakoupeny) a číselníky. Oproti zdrojovým databázím, které zůstávají stále primárním zdrojem těchto dat, jsou data v ODS:

- vyčištěná, konzistentní a konsolidovaná,
- aktualizována s určitým zpožděním (záleží na integrační cestě), zpravidla ze zdrojové aplikace,
- poskytována cílovým aplikacím, které se původně dotazovaly do zdrojových aplikací, což snižuje zátěž zdrojových systémů,
- ukládána bez historie (historické údaje jsou odmazány nebo zneplatněny) - s historií lze data dohledat v DWH,
- jsou uložena v jiné struktuře, [NOVOTNÝ, 2005] ji výstižně nazývá subjektově orientovanou. To znamená, že veškerá data o zákazníkovi jsou uložena v entitě zákazník, nikoli podle toho, která aplikace je v jaké struktuře poskytla.

Především subjektově orientovaný přístup dovoluje vytvořit jednotné rozhraní ODS pro všechny aplikace, což v praxi znamená např. službu GetCustomer, která vrátí veškerá konsolidovaná data o zákazníkovi bez ohledu na to, která z těch dat aplikace aktuálně

potřebuje (to je nevýhoda). Zároveň ale aplikace nemusí pátrat po x různých aplikacích po datech o zákazníkovi a dostane je aktuální z jednoho centrálního místa.

Data v ODS jsou pro aplikace určena pouze pro čtení (jelikož ODS není primárním vlastníkem těchto dat, tím je zdrojová aplikace), pouze zdrojové aplikace mají právo data v ODS aktualizovat.

Na teoretické úrovni se ODS zabývá např. [NOVOTNÝ, 2005], praktické modely na implementaci ODS jsou popsány v [REILLY, 2004] a [TMF, 2003] (primárně pro sektor telekomunikací). [REILLY, 2004] se rovněž pokouší definovat i univerzální rozhraní v Javě, které však v současnosti ještě není kompletní. V praxi jsem se podílel na implementaci ODS podle [REILLY, 2004].

Data poskytovaná ODS aplikacím by principiálně mohla být poskytována EAI nástrojem, avšak existují případy, kdy je výhodnější využít ODS:

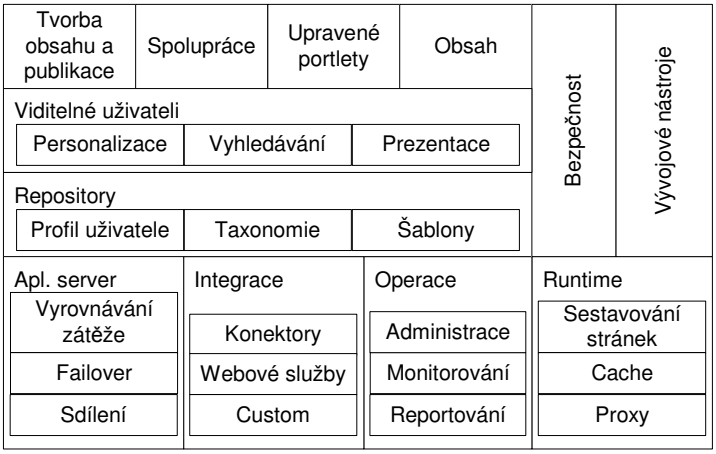
1. Velký objem předávaných dat (hranici současných EAI nástrojů tvoří řádově desítky kB) během 1 přenosu.
2. Zpracování dat nemusí být provedeno v reálném čase (tzv. dávkové).
3. Opakované žádosti o stejná data (ODS tedy funguje jako datová cache - tuto funkci jsou schopny některé EAI nástroje poskytnout).
4. Požadovaná transformace dat (zdrojová aplikace poskytuje data v odlišném formátu než je vyžadováno okolními aplikacemi) a není možné ji provádět on-line.
5. Dočasné řešení stávajících rozhraní při nahrazování zastaralé aplikace (integrované na datové úrovni) novější aplikací.
6. Požadovaná dostupnost dat v čase se liší od dostupnosti zdrojové aplikace.

12.3 EIP

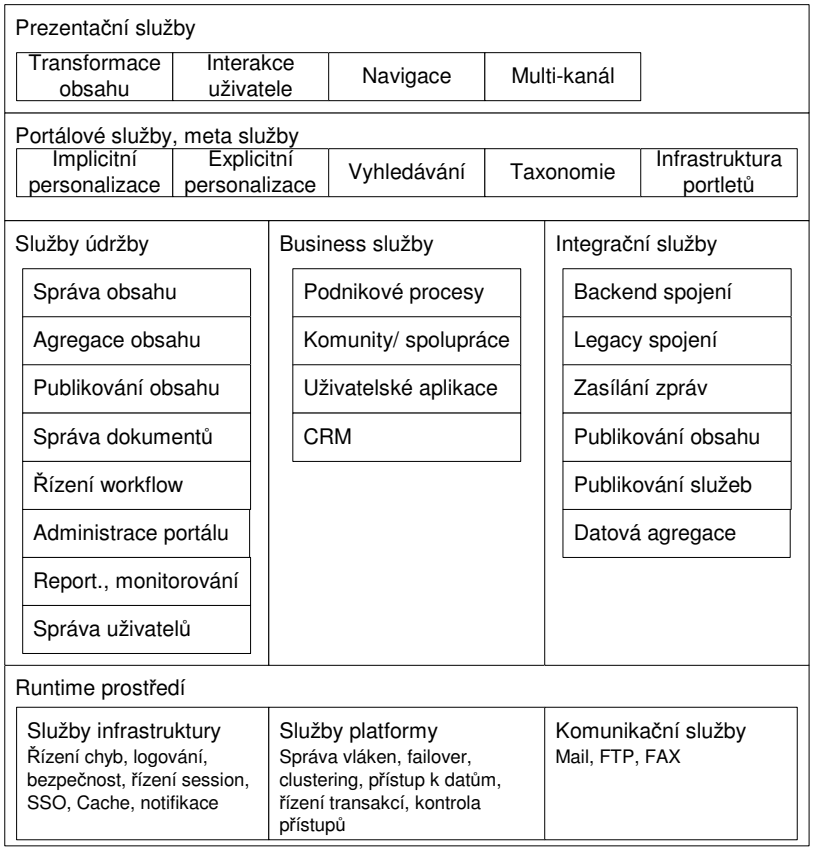
Rovněž u podnikového informačního portálu si popíšeme jeho základní komponenty. Jejich popis bude stručnější, neboť pouze relativně část funkcionality EIP má vztah k integraci aplikací. Zbývající funkcionalita tvoří předpřipravenou funkcionalitu pro webové aplikace, které se vyvíjejí přímo v portálovém prostředí. Při sestavování modelu EIP jsem opět nejdříve zanalyzoval dostupné informační zdroje - viz tabulka 17 - Přehled modelů EIP v informačních zdrojích (zdroj: autor). Vedle těchto modelů jsou stejně jako u BPI k dispozici modely dodavatelů EIP.

Identifikace informačního zdroje	Pokrytá oblast
[VALDES, 2003]	viz obrázek 42 - Model podnikového portálu (zdroj: [VALDES, 2003])
[HELLER, 2003]	viz obrázek 43 - Model podnikového portálu (zdroj: [HELLER, 2003). Na detailní úrovni se zabývá srovnáním portálu v J2EE a .NET prostředí.
[META, 2003]	viz obrázek 44 - Model podnikového portálu (zdroj: [META, 2003])
[PUSCHMANN, 2004]	viz obrázek 45 - Architektura procesního portálu (zdroj: [PUSCHMANN, 2004])

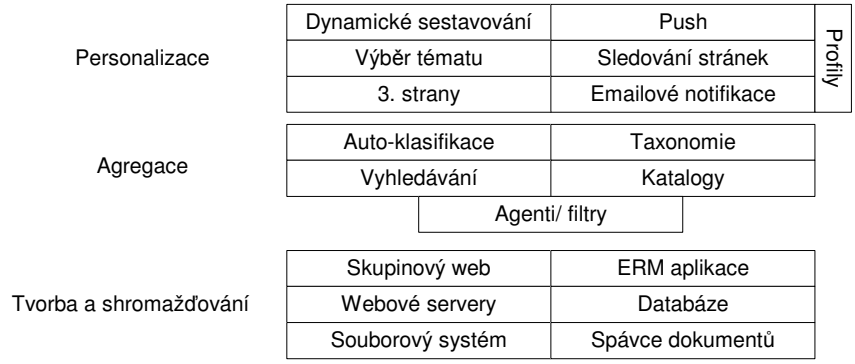
tabulka 17 - Přehled modelů EIP v informačních zdrojích (zdroj: autor)



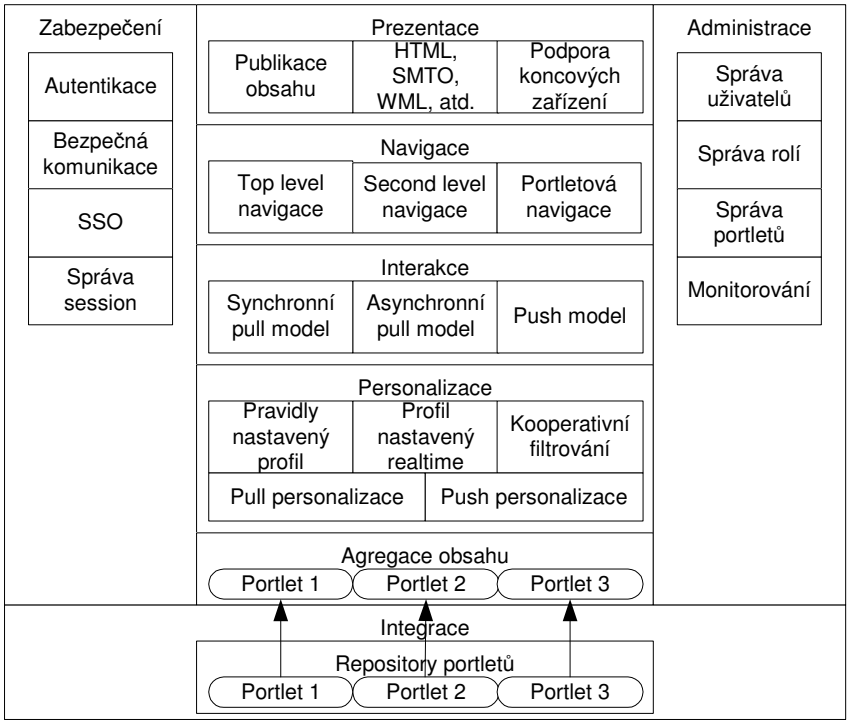
obrázek 42 - Model podnikového portálu (zdroj: [VALDES, 2003])



obrázek 43 - Model podnikového portálu (zdroj: [HELLER, 2003)]



obrázek 44 - Model podnikového portálu (zdroj: [META, 2003])



obrázek 45 - Architektura procesního portálu (zdroj: [PUSCHMANN, 2004])

Na obrázku 46 - Funkční model EIP (zdroj: autor) jsem znázornil skupiny hlavních funkčních komponent podnikového informačního portálu. Jsou to tyto:

A. Frontend rozhraní (externí rozhraní)

- Rozhraní pro poskytovatele obsahu - určeno pro firmy, které dodávají externí obsah pro portálové stránky. Typicky to mohou být marketingové firmy poskytující aktuální reklamní proužky.
- Rozhraní pro koncové uživatele - obsah portálu lze prezentovat nejen na webových klientech, ale i přes WAP na mobilních telefonech, přes IVR (Interactive Voice Response), email, atd. - těmto možnostem se říká portálové komunikační kanály. Tato komponenta zprostředkovává komunikaci přes komunikační kanály - pro webový přístup poskytuje webový server, pro mobilní přístup detekci typu telefonu apod.
- Rozhraní pro partnery - ne všechny výstupy musí mít podobu UI. Portál může typicky poskytovat webové služby (nebo jejich nestandardizovanou obdobu) přes toto rozhraní. Stejně tak může sloužit pro ověřování stavu konta na kreditní kartě při nákupu zákazníka.

- Administrační konzole - slouží pro přístup administrátora portálu a příslušnou administraci.

B. Obchodní komponenty

- Sledování uživatele - monitoruje chování uživatele na portálu, např. v jaké posloupnosti stránky portálu navštívil, jak dlouho se jim věnoval atd.
- Řízení marketingových kampaní - portálová řešení nabízejí širokou škálu cílených marketingových akcí, které lze definovat prostřednictvím této komponenty. Podle zadaných business pravidel, která pracují se všemi možnými údaji, které má portál k dispozici, mohou např. cíleně zobrazovat reklamní proužky, odesílat reklamní emaily, poskytovat slevy na zboží koupené přes portál apod.
- Produktový katalog a konfigurátor - komponenta určená pro elektronické obchody, pomocí které si zákazník vybere a případně nakonfiguruje požadovaný produkt. Zpravidla umožňuje integraci na externí databázi produktů, kterou má podnik již vytvořenou.
- Segmentace zákazníků - umožňuje roztřídit zákazníky podle business pravidel do definovaných segmentů, které jsou dále využívány např. pro různé slevy apod.

C. Jádro portálu

- Řízení uživatelského rozhraní - upravuje stav uživatelského rozhraní, definuje atributy jednotlivých stránek ve vztahu k jiným komponentám (např. pro řízení marketingových kampaní, personalizaci).
- Personalizace - umožňuje podniku definovat chování uživatelského rozhraní dle atributů konkrétního zákazníka. Např. skrývá některé části obrazovky podle toho, v jakém segmentu zákazník je, nebo dle jiných business pravidel.
- Kustomizace - umožňuje uživateli upravit si uživatelské rozhraní dle svých individuálních potřeb (např. změnit barvy, schovat nepotřebné části obrazovky).
- Spolupráce v komunitách - podporuje komunikaci mezi uživateli portálu, např. formou diskuzních fór, sdílených obrazovek na kreslení.
- Správa obsahu - spravuje repository se statickým obsahem stránek, který je uložen v databázi (může být uložen i formou souborů) a poskytuje jej pro generování stránek. Tato repository může být umístěna i v externím systému (tyto systémy se označují CMS (Content Management System)) a pak se k ní přistupuje prostřednictvím speciálního adaptéru do repository obsahu.
- Řízení posloupnosti stránek - definuje posloupnost, v jakém jsou stránky uživateli zobrazovány s využitím business pravidel (např. pokud zákazník patří do VIP segmentu, pak je mu při potvrzení objednávky zobrazena exklusivní nabídka na další zboží, zatímco ostatním segmentům nikoli).
- Správa instancí - řídí a spravuje instalaci instancí portálu.
- Vyhledávání - prohledává obsah portálu - zpravidla statický obsah, tedy intenzivně spolupracuje se správou obsahu.
- Správa uživatelů - zajišťuje registraci nových uživatelů portálu a aktualizaci údajů o uživatelích.
- Jednotné přihlášení - pokud portál umožňuje odskoky do externích webových aplikací s povinným přihlášením, pak tato komponenta zajišťuje sdílení přihlašovacích práv s těmito aplikacemi.
- Řízení bezpečnosti - zajišťuje bezpečnostní mechanismy portálu (např. zabezpečený přenos dat, omezený přístup z vybraných IP adres, přístup ke stránkám dle práv uživatele, přihlášení do portálu).
- Správa přístupu - definuje přístupová práva uživatelů k jednotlivých stránkám.

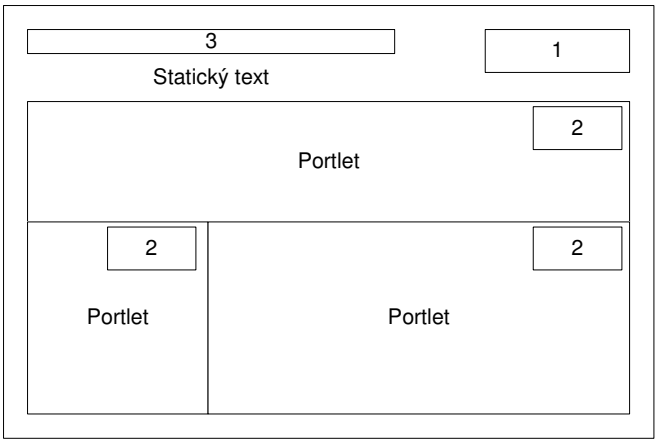
D. Interní rozhraní

- Adaptér do repository obsahu - připojuje interní nebo externí repository se statickým obsahem stránek.
- Technologické adaptéry - umožňuje přístup k podnikovým aplikacím skrze standardní technologie, obdoba technologických adaptérů BPI.
- Aplikační adaptéry - umožňuje přístup k podnikovým aplikacím skrze jejich specifická rozhraní.
- Adaptér do portálové databáze - připojuje portálovou databázi, která zpravidla obsahuje konfigurační nastavení portálu, jeho uživatelů (zejména personalizační a kustomizační údaje) a údaje o uživatelích.

Rozhraní pro poskytovatele obsahu	Rozhraní pro koncové uživatele	Rozhraní pro partnery	Administrační konzole	Frontend rozhraní
Sledování uživatele	Řízení marketingových kampaní	Produktový katalog a konfigurátor	Segmentace zákazníků	Business komponenty
Řízení uživatelského rozhraní	Personalizace	Kustomizace	Spolupráce v komunitách	
Správa obsahu	Řízení posloupnosti stránek	Správa instancí	Vyhledávání	Jádro portálu
Správa uživatelů	SSO	Řízení bezpečnosti	Správa přístupů	
Adaptér do repository obsahu	Technologické adaptéry	Aplikační adaptéry	Adaptér do portálové databáze	Backend integrace

obrázek 46 - Funkční model EIP (zdroj: autor)

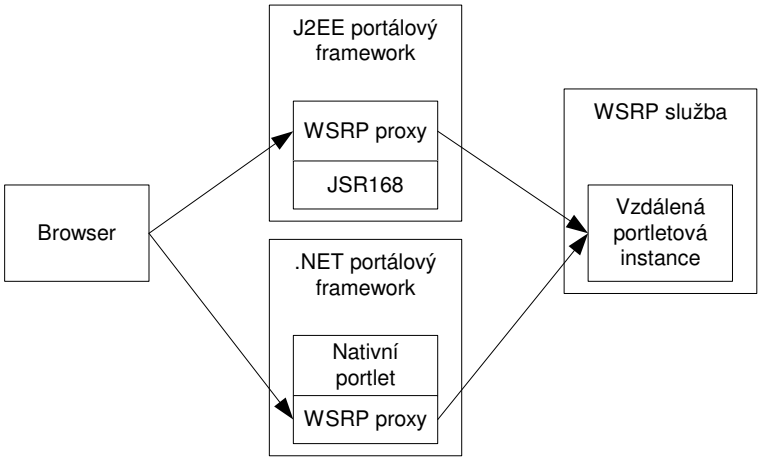
Komunikační model EIP je poměrně jednoduchý. Klient EIP vyšle požadavek na zobrazení konkrétní stránky, požadavek je zpracován a předán webovým serverem do jádra portálu. Podle parametrů stránky (jsou zpravidla definovány příznaky, tzv. tagy, v těle stránky) jsou požádány jednotlivé komponenty portálu o poskytnutí vstupů pro tuto stránku. Hlavní část stránky poskytuje správce obsahu, portálová databáze a údaje z podnikových aplikací získané přes adaptéry. Poté je vygenerovaná stránka předána zpět klientovi. Podkladovou platformou portálového frameworku je aplikační server (AS).



obrázek 47 - Schéma prezentační vrstvy portálu (zdroj: autor)

Obrázek 47 - Schéma prezentační vrstvy portálu (zdroj: autor) zobrazuje typickou portálovou stránku. Obsahuje několik prvků:

- Bod 1 - Ovládací prvky portálu, zpravidla kustomizační tlačítka (volba barevného schéma, jazyka, změna uživatelských údajů apod.), odhlašovací tlačítko.
- Bod 2 - Ovládací prvky portletu, např. skrytí/zobrazení portletu, výběr viditelných prvků portletu.
- Bod 3 - Uživatelské menu pro výběr dalších stránek.
- Statický text - narozdíl od obsahu portletů se nemění.
- Portlet - autonomní část obrazovky, která z hlavní části přejímá potřebné údaje o uživateli; může být generována přímo portálem nebo vzdáleně - viz obrázek 48 - Integrace přes vzdálené portlety (zdroj: autor).



obrázek 48 - Integrace přes vzdálené portlety (zdroj: autor)

12.4 ETL

Funkcionalitou nástrojů ETL i postupem jejich implementace se detailně zabývá [NOVOTNÝ, 2005], proto zde pouze shrňme základní funkcionalitu ETL:

- Definice vstupních a výstupních datových zdrojů - primárně se pracuje s SQL databázemi a textovými soubory.
- Definice transformačních pravidel - mezi transformační úlohy patří:
  - transformace dat,
  - spuštění programu,
  - spuštění SQL příkazu,
  - spuštění jiné ETL.
- Definice průběhu transformace - zejména nastavení časového plánu spouštění ETL, reakce na chyby při transformacích, nastavení transakcí.

12.5 XQuery Gateway

[KERNOCHAN, 2003] charakterizuje tyto produkty (nazývá je EII (Enterprise Information Integration) produkty) základními komponentami:

- Adresář (též data dictionary dle [FOURNIER-MOREL, 2004]) - obsahuje informace (metadata) o datech v připojených databázích.
- Konverzní modul - slouží pro konverzi příchozích i odchozích dat, např. z nerelačního do relačního formátu nebo z relačního do objektového formátu. A to dle požadavků cílových aplikací i zdrojových aplikací (pokud se např. provádí update zdrojových dat).
- Externí rozhraní (frontend) - produkt se navenek vůči aplikacím i uživatelům (databázovým administrátorům, vývojářům) musí tvářit jako standardní databáze, tj. s podporou SQL a transakcí.

- Interní rozhraní (backend) - slouží pro komunikaci s integrovanými datovými zdroji. Do základní podporované skupiny patří:
  - relační databáze (Oracle, MS SQL Server, DB2, ...),
  - LDAP,
  - textové zdroje (soubory typu XML, CSV, dokumenty...).
  - někteří dodavatelé (např. BEA) nabízejí možnost připojení datových zdrojů skrze webové služby.

Vedle základních komponent se můžeme setkat i dalšími komponentami:

- datová cache (interní operační datový sklad),
- optimalizace dotazů,
- administrační konzole,
- tvůrce datových náhledů (vývojový nástroj pro tvorbu náhledů pro externí rozhraní) - praktickou ukázku nalezneme v [TMR02].

Podobně charakterizuje EII produkty také [IPEDO, 2005].

Komunikace probíhá následovně - volající aplikace zašle žádost o data (ve formě SQL nebo XQuery), XQuery Gateway žádost zanalyzuje a pomocí adresáře (nebo přímo) vyhledá potřebné zdroje. Původní požadavek rozdělí do dotazů pro informační zdroje a odešle jej. Příchozí odpovědi agreguje, zformátuje a konsolidovaný výstup předá volající aplikaci.

13. Rozšířený model komplexní integrované architektury PIS

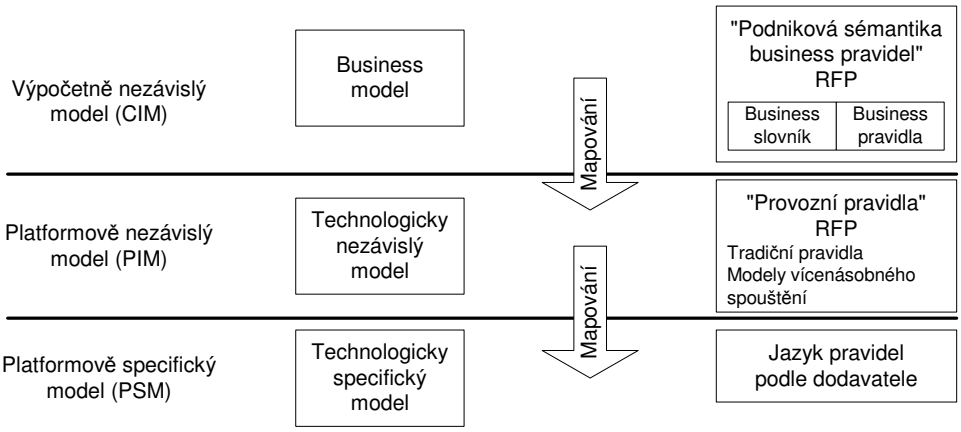
V kapitolách 11 Model komplexní integrované architektury PIS a 9.5 Modelování procesů jsem se zmínil o zpracování business pravidel. Dosud jsme předpokládali umístění komponent pro zpracování business pravidel (business rules) uvnitř integrační platformy. Tato komponenta je explicitně popsána v modelu integračního serveru. I ostatní integrační komponenty zpracovávají business pravidla, ale nemají na to samostatné komponenty - tyto operace jsou uloženy přímo ve zdrojovém kódu. V této kapitole si stručně popíšeme tyto komponenty, pokud jsou umístěny vně integračního serveru, a nastíníme si možnosti jejich využití pro další integrační komponenty.

13.1 Business rules engine

[DIAMONDCLUSTER, 2001] definuje BRE jako nástroj, který "odděluje obsah od zpracování pro zjednodušení vývoje a údržby. Pravidla mohou být definována v přirozeném jazyce, aby umožnily obchodním složkám definovat business pravidla. Tím mohou být pravidla měněna v kratším čase."

[MCCOY, 2003b] definuje BRE jako "prostor pro spouštění pravidel, kde pravidla jsou dostupná méně zkušeným technickým vývojářům, aby podporovali takřka v reálném čase změny business pravidel."

Konceptuálním modelováním business pravidel se zabývá [OMG] - viz obrázek 49 - Modelování business pravidel dle OMG (zdroj: [LINEHAN, 2005]).



obrázek 49 - Modelování business pravidel dle OMG (zdroj: [LINEHAN, 2005])

Definicí, co to business pravidla jsou, se detailně zabývá [BAJEC, 2005b], odkud vybírám následující definici: "Business pravidlo je výrok, který definuje nebo omezuje některý aspekt podnikání. Je určen pro uplatnění podnikové struktury nebo ke kontrole nebo vlivu na chování podnikání." Pravidlo je reprezentováno ve formě IF podmínka THEN akce1 ELSE akce2. [ROSS, 2003] popisuje detailně konkrétní vzory business pravidel. Pravidla mohou být sdružována v tzv. politikách (např. pro lepší orientaci).

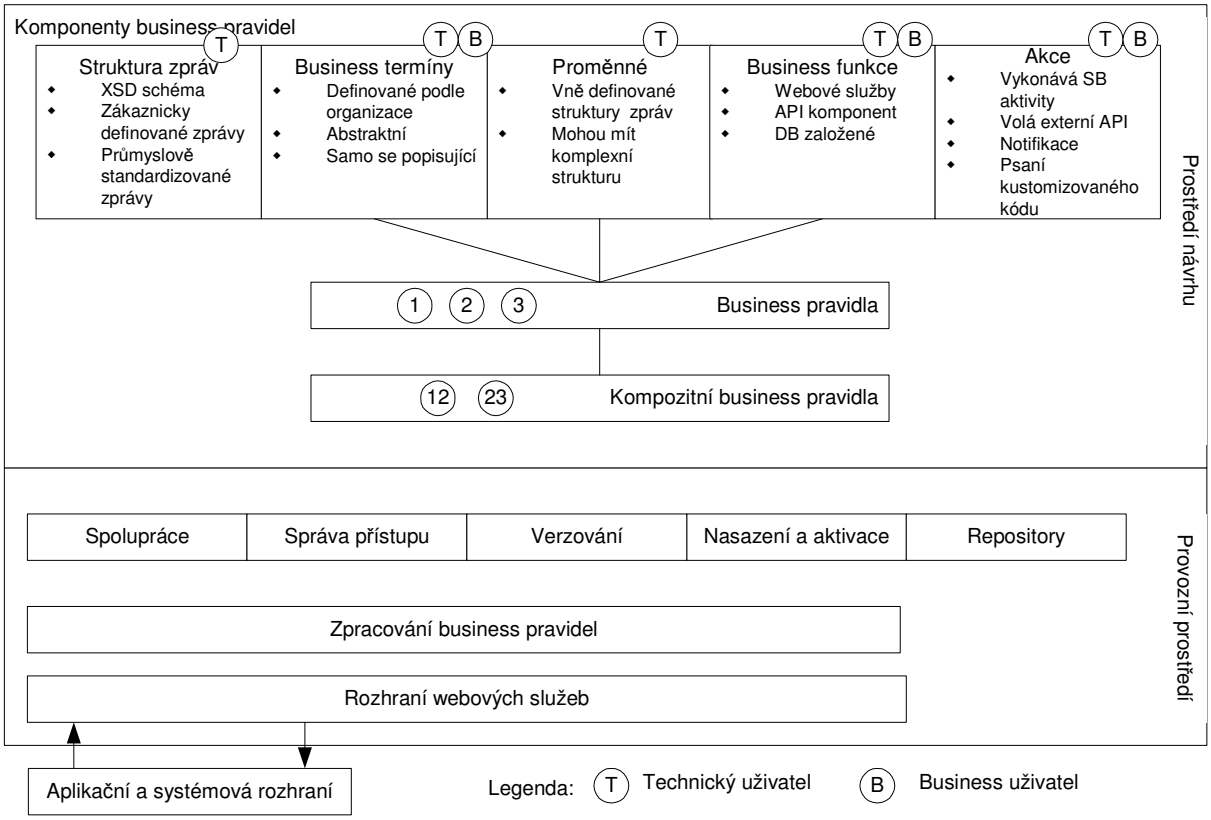
Využití BRE je široké - jejich nativní schopností je uplatnit se v kterékoliv podnikové aplikaci, neboť z principu nahrazují pravidla zakódovaná v aplikacích. [TIBCO, 2005] shrnuje problémy, které se BRE snaží řešit:

- Automatizace aplikování business pravidel.
- Společná repository pro důležitá business pravidla.
- Zabezpečení, že všechny potřebné informace jsou dostupné před vlastním rozhodnutím.
- Agregování a hledání smyslupnosti informací v podniku.

Pro hlubší poznání možností a struktury BRE odkazuji na [BMKM2], [KOVACIC, 2002], [ROSS, 2003] a [ARLOW, 2003]. Spolupráce aplikace a BRE funguje jednoduše -

aplikace zpracovává svůj vlastní kód a v daném místě zavolá vzdálenou službu BRE se vstupními parametry a identifikací pravidla, BRE pravidlo vyhodnotí a výsledek předá zpět aplikaci, která pokračuje dále ve zpracovávání.

BRE mají své vývojové prostředí pro definování pravidel a provozní prostředí pro samotný běh zpracovávání pravidel. Obrázek 50 - BRE framework (zdroj: [ROFAIL, 2005]) znázorňuje základní architekturu BRE.



obrázek 50 - BRE framework (zdroj: [ROFAIL, 2005])

Způsobem vyhledávání business pravidel v rámci projektu se zabývá např. [ILKAEV], [ROSS, 2003] a [HALLE, 2002].

13.2 BRE v integračním projektu

Nyní se podívejme na vztah business pravidel, resp. BRE a integrační platformy. Přední dodavatelé integračních serverů poskytují ve svých produktech nástroj typu BRE i možnost připojení externího BRE. Mezi typické využití business pravidel při integraci patří:

- Spuštění integračního procesu (např. pokud KPI klesne pod určitou úroveň).
- Rozhodování v integračním procesu (jestliže platí A, pak se vydej cestou X, jinak cestou Y).
- Monitorování integračních procesů (notifikace uživatele při stavu 2).

Jak jsme si řekli v kapitole 9 Integrační proces a integrační případ, tak integračním procesem v širším pojetí chápeme komunikaci mezi aplikacemi nejen přes integrační server, ale přes jakoukoliv integrační komponentu. V kombinaci se schopnostmi BRE tak uvažujme dále o tom, zda je možné **rozšířit uplatnění BRE i na další integrační komponenty**. Z principu se zdá, že ano, neboť BRE je adaptabilní na jakýkoliv typ aplikací.

Představme si tedy pravidlo, které bude určovat, do jakého zákaznického segmentu zákazník patří. Toto pravidlo si implementujme do BRE a zkoumejme jeho možné využití v integračních komponentách:

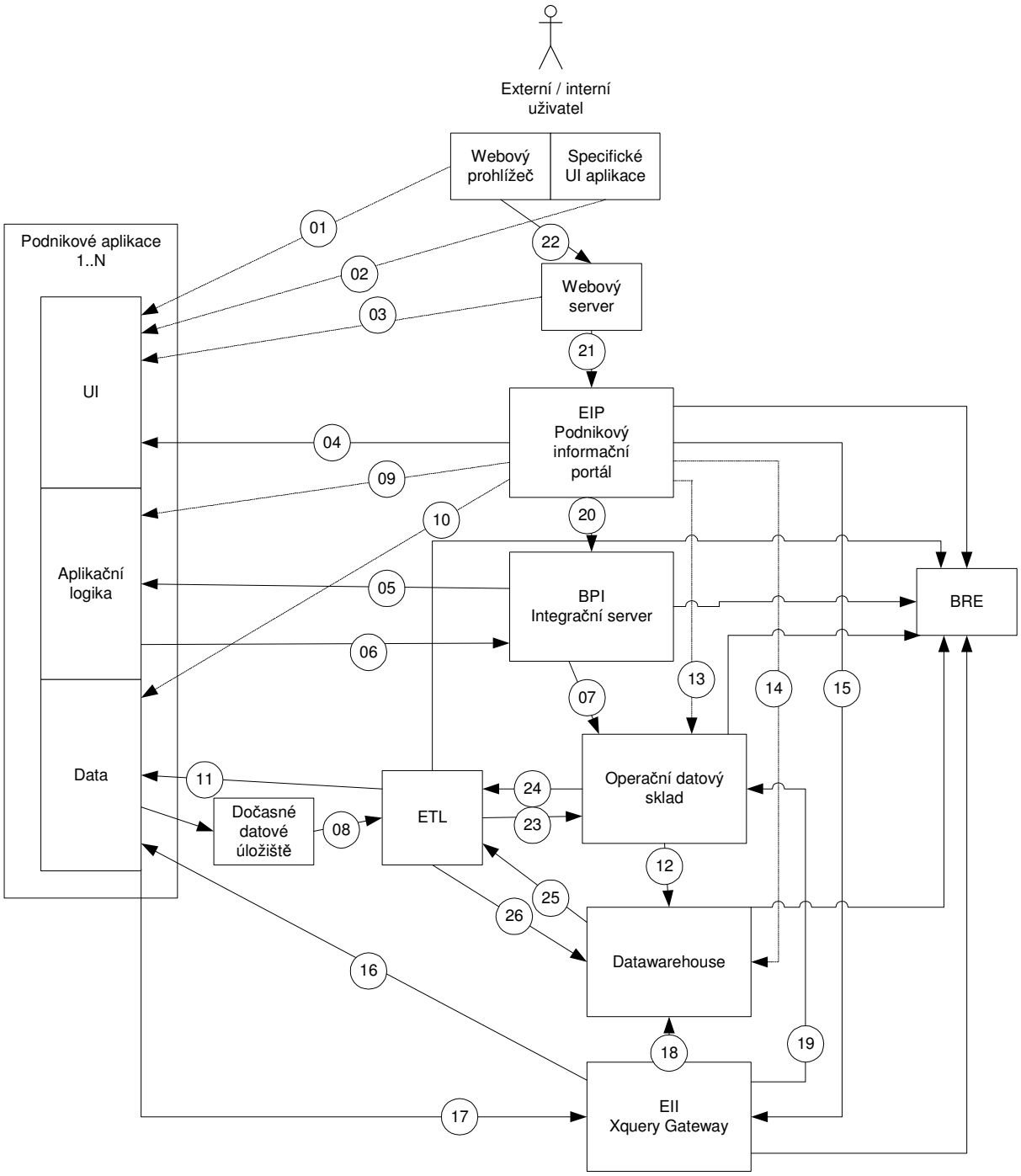
1. EIP - pravidlo využijeme např. pro zobrazení personalizované stránky, která je určena pouze pro zákazníky z určitého segmentu. Využití BRE v EIP nabízí např. BEA Weblogic Portal.
2. BPI - pravidlo se využije např. při větvení procesů realizace objednávky dle zákaznického segmentu.
3. ODS, DWH - využijí pravidlo při určování zákaznického segmentu při dotazu integrované aplikace na informace o zákazníkovi.
4. XQuery Gateway - pravidlo se aplikuje při dotazu integrované aplikace na zákaznický segment zákazníka (bude na něj odkazováno z náhledu na zákazníka).
5. ETL - může pravidlo využít při transformaci zákaznických dat, pokud bude potřebovat určit do jakého zákaznického segmentu zákazník patří.

Bez využití sdíleného BRE pro tyto komponenty to znamená naprogramovat tuto funkcionalitu v této variantě 6 krát. Pokud to vynásobíme počtem definovaných pravidel, pak se dostáváme k počtu pravidel, která není možné efektivně udržovat konzistentní. K dalším chybám PIS bude docházet díky rozdílným časům implementace těchto pravidel do integračních komponent.

S reálným příkladem jsem se setkal v reálném projektu (viz kapitola 17.6.4 Projekt 4 - Budování ODS (2005 - 2006)), kde bylo třeba současně vyvinout dvě integrační cesty, z nichž jedna byla záložní. Jedna byla realizována prostřednictvím integračního serveru, druhá pomocí ODS. Obě cesty musely implementovat identická pravidla, kterých bylo cca 300. Nebylo využito sdíleného BRE, což vedlo k duplicitní práci od návrhu, vývoje až po testování. V průběhu testování se obě cesty porovnávaly, což bylo velmi pracné a vedlo to k opakujícím se chybám. Výsledná pracnost implementace těchto pravidel tedy nebyla jen dvojnásobná (pro dvě integrační komponenty), ale ještě větší, neboť se musela zajistit konzistence obou cest.

Obrázek 51 - Rozšířený model komplexní integrované architektury PIS (zdroj: autor) zobrazuje umístění sdíleného BRE v celkové koncepci integrované architektury PIS. V praxi jsem se s využitím sdíleného BRE dosud nesetkal, ale z hlediska technologie to nepovažuji za neřešitelný problém, neboť EIP a EAI komponenty s BRE již samostatně spolupracují. U databázově orientovaných integračních komponent jsem možnost využívat externí BRE v informačních zdrojích nenalezl, přesto se domnívám, že prostřednictvím databázové procedury či webové služby by funkcionalita BRE mohla být zpřístupněna i těmto integračním komponentám. Zde vidím prostor pro další zkoumání s reálnou přidanou hodnotou pro podnik.

BRE chápejme jako názorný příklad, jakým způsobem efektivně využít prezentovaný model. Tento příklad dokumentuje způsob, jakým nás model **vede k aplikaci dalších sdílených komponent v integrované architektuře PIS**. Vedle BRE bychom se dále mohli dále zabývat implementací sdílené společné repository metadat či zabezpečovací vrstvy (např. SSO) v podobném kontextu.



obrázek 51 - Rozšířený model komplexní integrované architektury PIS (zdroj: autor)

14. Výběr integrační platformy

Tato kapitola se zabývá procesem výběru nejvhodnějšího integračního produktu pro realizaci integrační platformy. Proces je detailně popsán, včetně doporučených kritérií (v příloze) výběru a jejich váh podle typu integračního projektu, pro integrační komponentu typu integrační server. Pro ostatní integrační komponenty jsou uvedeny odkazy na informační zdroje.

Výběrem integrační platformy se přirozeně zabýváme pouze u projektů, jejichž součástí je kompletní implementace integrační platformy. Při jejím výběru vycházíme z integrační strategie. Podle integrační strategie určíme, jakou kategorii integračních produktů budeme potřebovat:

1. databázi pro realizaci ODS či DWH
2. portálový framework pro budování podnikového portálu
3. integrační server pro integraci aplikací v reálném čase
4. ETL pro dávkovou integraci databází
5. XQuery Gateway pro integraci informačních zdrojů v reálném čase.

Tento proces probíhá ve fázi úvodní studie, která identifikuje a zhodnotí jednotlivé produkty ve vybrané kategorii. Zvažuje-li podnik možnost realizovat projekt vlastními zdroji (tzv. in-house), provádí tento výběr sám, pokud se předpokládá realizace projektu externí společností, je podnik od tohoto výběru částečně odstíněn, neboť za vhodný výběr odpovídá externí integrátor.

Avšak vzhledem k tomu, že integrační platforma se v případě úspěšné implementace stává vysoce kritickou aplikací pro podnik, je přinejmenším vhodné, aby se odpovědní zástupci podniku informovali o vlastnostech integrační platformy. Posuzování platformy by měla vždy provádět osoba s širokými technologickými znalostmi, detailní znalostí podnikových aplikací a procesů, obeznámená s plánovaným rozvojem podnikového informačního systému.

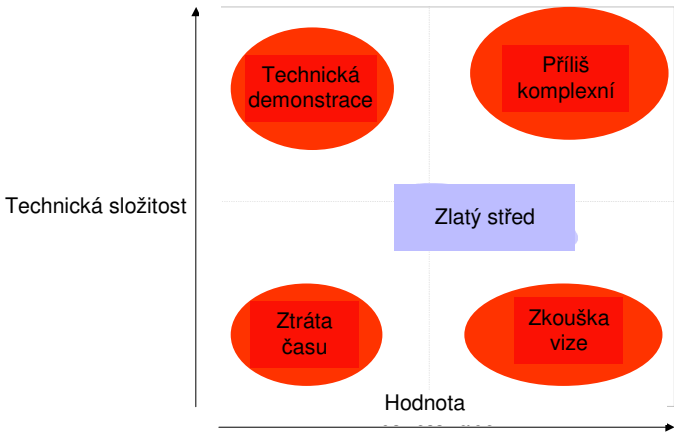
My dále omezíme popis výběru integrační platformy na výběr integračního serveru, protože tuto kategorii považuji za nejsložitější. Při výběru vhodné databáze odkazuji na [FRENTZEN, 2003]. Při výběru produktu pro EIP doporučuji se soustředit na funkční portálové komponenty, neboť integrační komponenty při výběru portálu nehrají rozhodující roli (tržní produkty se odlišují především v poskytované funkcionalitě, nikoli v možnostech integrace na interní aplikace). Pravidelně jsou k dispozici srovnávací studie EIP produktů - viz např. [MURPHY, 2002] a [META, 2003]. Při výběru ETL doporučuji se informovat v [NOVOTNÝ, 2005] a následně posuzovat přidanou funkcionalitu produktů. Trh s XQuery Gateway produkty nepovažuji ještě za dostatečně stabilizovaný (velcí hráči předvádějí nebo připravují první verze produktů), proto nelze výběr tohoto produktu ještě doporučit. Pro tento typ produktů bude rozhodující, jaké informační zdroje je produkt schopen integrovat, jak efektivně lze vytvářet konsolidované náhledy na tyto zdroje a především doba odezvy klíčových náhledů. V každém případě je nutné požádat dodavatele o POC a až podle výsledků se rozhodnout. Srovnání produktů XQuery Gateway nalezneme v [KERNOCHAN, 2003].

14.1 Proces výběru

Proces výběru integrační platformy probíhá v několika krocích v interakci mezi zákazníkem a potenciálním dodavatelem a integrátorem - iniciátorem je vždy zákazník (volně přeloženo z [HERRERA, 2003]):

1. Žádost o informace (RFI - Request For Information) - základní informace o produktu, nejlépe srovnání s dalšími produkty zpracované renomovanou firmou.

2. Žádost o předvedení (RFD - Request For Demonstration) - předvedení základních funkcí a schopností produktu.
3. Žádost o nabídku (RFP - Request For Proposal) - detailní informace o produktu, dodavateli a jeho partnerech (poskytuje zejména integrátor).
4. Výběr produktu - toto je popsáno detailně dále.
5. Zkouška koncepce (POC - Proof Of Concept) - zkouška produktu na malé části business požadavků, obrázek 52 - Zaměření POC dle Gartner (zdroj: [GROENENDAAL, 2002]) demonstruje doporučené pojetí POC.
6. Žádost o cenovou kalkulaci (RFQ - Request For Quotation) - tato část je většinou součástí RFP, zde může následovat upřesnění
7. Žádost o smlouvu (RFB - Request For Bid) - uzavření smlouvy



obrázek 52 - Zaměření POC dle Gartner (zdroj: [GROENENDAAL, 2002])

Požadavky kladené na integrační platformu jsou vždy závislé na specifických integračních potřebách daného podniku, přesněji řečeno na jeho podnikového informačního systému a na jeho předpokládaném strategickém rozvoji. Nelze tedy obecně prohlásit některou platformu za univerzálně nejlepší. V příloze 17.5 Kritéria výběru integračního serveru je k dispozici detailní přehled kritérií (zejména technologických), která lze využít jako vodítko při výběru integrační platformy. Hlavním zdrojem pro tato kritéria byla kritéria zpracovaná v [OVUM, 2001], dále informace v kapitole 12.1 Integrační server a obchodní materiály dodavatelů integračních platforem BEA Systems ([BEA WebLogic], [GANDHI, 2005], [BEA AquaLogic], [CHIKARMANE, 2005]), Tibco ([TIBCO, 2003a], [TIBCO, 2004], [TIBCO, 2003b]), Microsoft ([JURÍK, 2003]). Zároveň jsem do kritérií doplnil o praktické poznatky z integračních projektů, na kterých jsem se podílel. Porovnáním integračních produktů se zabývá např. [ENDLICH, 2004]. Oproti [OVUM, 2001] jsem zavedl logická uskupení kritérií, doplnil chybějící kritéria, navázal kritéria na jednotlivé úrovně integrace a popsal jednotlivé kroky výběru integrační platformy. Naopak jsem nepřevzal bodové ohodnocení kritérií popsané v [OVUM, 2001], neboť toto je vždy individuální dle potřeb připravovaného projektu, a dále jsem vynechal kritéria specifická pro mezipodnikovou integraci (B2B), která je nad rámec této práce.

Samotný proces výběru doporučuji provádět v následujících krocích:

1. Rozhodnutí o požadované úrovni integrace - podle tabulky 6 - Integrační úrovně (zdroj: autor) se rozhodneme, jakou integrační úroveň od integrační platformy potřebujeme podporovat. U integračního serveru nemá smysl uvažovat prezentační integrační úroveň, proto se rozhodneme mezi
    a. datovou (D),
    b. služeb (S),

c. procesní (P).

Mezi úrovněmi integrace je hierarchický vztah v tom smyslu, že integrace služeb vyžaduje integraci datovou (v rámci služeb se předávají data) a integrace procesní vyžaduje integraci datovou i integraci služeb (v rámci procesů se předávají data i volají služby). Podle požadované úrovně je poté možné vyřadit některá kritéria, která jsou např. požadována pouze pro procesní integraci, pokud není vyžadována. Pro usnadnění je u každého kritéria uvedeno, pro jakou úroveň integrace je relevantní.

2. Rozhodnutí o primárním využití integrační platformy - integrační platforma je zpravidla implementována v rámci určitého typu projektu tak, jak byly vymezeny v kapitole 2.2 Typy projektů. Platforma musí plně podporovat tento projekt, proto je třeba přiřadit určitým skupinám kritérií (dle tabulky 18 - Primární kritéria výběru integrační platformy dle typu projektu (zdroj: autor)) vyšší prioritu.

Typ integračního projektu	**Popis**	**Primární kritéria**
Agregační	Integrace nové aplikace s mnoha různými stávajícími aplikacemi, kde dochází k agregaci mnoha zdrojů (dat a služeb) do jednoho. Typickým příkladem je implementace BPI pro podnikový informační portál.	Architektura - Message broker Architektura - Message broker Architektura - Aplikační a technologické adaptéry Architektura - Zabezpečení Vývoj - Vývojové nástroje Vývoj - Testovací nástroje Zavádění a provoz - Administrace Zavádění a provoz - Monitorování
Migrační	Pročišťování IT infrastruktury, nebo migrace na vyšší verze produktů, kde dochází k normalizaci rozhraní, zavádění standardů, odstraňování nadbytečných rozhraní.	Architektura - Aplikační a technologické adaptéry Architektura - Business rules engine Architektura - Repository metadat Architektura - Otevřenost Vývoj - Normalizace Zavádění a provoz - Monitoring
Procesní	Automatizace podnikových procesů, kde hlavní důraz je kladen nikoli na aplikace, ale na vztahy mezi nimi. Ovlivňuje všechny integrované aplikace i uvnitř.	Architektura - Aplikační a technologické adaptéry Architektura - Business rules engine Architektura - Repository metadat Architektura - Otevřenost Architektura - Transakční monitor Architektura - Řízení událostí Architektura - BPM Architektura - BAM Vývoj - Normalizace Vývoj - Vývojové nástroje Vývoj - Modelování procesů Vývoj - Testovací nástroje Zavádění a provoz - Administrace Zavádění a provoz - Monitorování Rozšiřitelnost

tabulka 18 - Primární kritéria výběru integrační platformy dle typu projektu (zdroj: autor)

3. Nastavení preferencí jednotlivých kritérií - dle specifického projektu (preferovaných standardů, integrovaných aplikací, nároků na další rozvoj apod.) je nastavena důležitost jednotlivých kritérií.
4. Analýza nabízených produktů - jednotlivým kritériím jsou přiřazovány dostupné konkrétní informace o produktech a na základě těchto informací je přiřazeno číselné ohodnocení produktu pro dané kritérium dle stanovené škály. Pokud je to možné, podnik by měl posléze získat od dodavatele vybrané integrační platformy písemnou garanci pravdivosti informací u všech klíčových kritérií, nejlépe se specifikací konkrétního užití dané vlastnosti produktu v projektu.

   Kritéria jsou zaměřena na funkcionalitu zkoumaného produktu dle obrázku 35 - Model integračního serveru (zdroj: autor) podporující všechny úrovně integrace a jsou rozčleněna do následujících hlavních skupin:

Skupina kritérií	Popis skupiny
Identifikace produktu	Základní údaje o produktu.
Architektura	Charakteristika jednotlivých komponent produktu.
Vývoj	Parametry ovlivňující náročnost kustomizace produktu a doplňkový vývoj.
Zavádění a provoz	Parametry ovlivňující nasazení a provozní využívání produktu.
Rozšiřitelnost	Vazby na další produkty, předpokládaný rozvoj celého řešení.
Cena	Cenové náklady řešení.

tabulka 19 - Skupiny kritérií výběru (zdroj: autor)

   Skupiny Architektura, Vývoj, Zavádění a provoz, Rozšiřitelnost kopírují životní cyklus integrační platformy od analýzy a návrhu (Architektura), implementace (Vývoj) k zavádění, provozu a údržbě (Zavádění a provoz, Rozšiřitelnost). Zbývající dvě skupiny Identifikace produktu a Cena jsou obchodního rázu. Kritéria jsou uvedena v tabulce 25 - Kritéria výběru integračního serveru (zdroj: autor a [OVUM, 2001]). Kritéria uvedená ve skupině Architektura využívají model komponent integrační platformy prezentovaný v kapitole 12.1 Integrační server.
5. Vyhodnocení váženého aritmetického průměru všech kritérií získáme porovnání jednotlivých produktů a zvolíme nejvhodnější produkt.

15. Testování

Kapitola Testování v první části popisuje obecnou typologii testů a specifikuje tak obsah testovací fáze. S pomocí vytvořeného modelu identifikuje obsahové odlišnosti testování v integračních projektech. V druhé části představuje obecný proces testování a následně zkoumá odlišnosti testovací fáze v integračních projektech z procesního hlediska.

Cílem této kapitoly není pokrýt celý rozsah zajištění kvality (Quality Assurance (QA)) softwarových produktů, nýbrž se budeme zabývat pouze přípravou a samotným jednorázovým vyhodnocením kvality sledovaného produktu. Tento proces se označuje jako testování. [FRANK, 2002] uvádí dvě definice kvality dle ANSI/IEEE 828, 1990:

- úroveň, do jaké se systém, komponenta nebo proces shoduje se specifikovanými požadavky
- úroveň, do jaké se systém, komponenta nebo proces shoduje se zákaznickými nebo uživatelskými potřebami nebo očekáváními.

Kompletní proces zajištění kvality dle [FRANK, 2002] obsahuje tyto činnosti:

1. tvorba dokumentace kvality,
2. revize plánů, specifikací a záznamů,
3. audit procesů, procedur a vývojářů 3. stran,
4. monitorování plánů implementace, schůzek a status reportů,
5. inspekce dodaných komponent, kódů a dokumentace,
6. účast při revizi designových dokumentů,
7. nápravné akce - jejich identifikace a kontrola realizace,
8. asistování při výběru nástrojů, technik a metodik,
9. testování formou účasti při realizaci testů a jeho řízením.

Předmětem testování v integračních projektech je implementace integrační platformy do PIS, nikoli samostatná integrační platforma (její kvalita je zkoumána při jejím výběru - viz kapitola 14 Výběr integrační platformy). V této kapitole budeme hledat optimální způsob, jak testovat v integračních projektech. Při hledání způsobu testování vyjdeme ze standardních testů softwarových produktů a budeme je aplikovat na typickou architekturu PIS integrovaného integrační platformou.

15.1 Typologie testů

Testování je poměrně složitý proces, který vychází z předem definovaných požadavků na integrovaný systém a který začíná sestavením testovacího týmu, pokračuje přípravou testovacích plánů, přesnou definicí akceptačních kritérií v testovacích případech, přípravou samotného testování a končí prováděním testů, podpůrnou činností při analýze a opravě nalezených chyb, retestováním dodaných oprav a vyhotovením testovacího protokolu. Tento proces je popsán v dostupné literatuře (např. [FRANK, 2002]). Testování začíná již v průběhu vývoje, v tradičním pojetí se jedná o provádění tzv. unit testů (budou blíže definovány v dalším textu), existují však i postupy, kde je vývoj řízen přímo výsledky souběžně prováděných testů (např. Test Driven Development [HOHPE, 2002b], [GOLD, 2004]).

My se tímto procesem nebude detailněji zabývat a zaměříme se na přímo na obsah testování. Kompilací informačních zdrojů (např. [FRANK, 2002], [HOHPE, 2002b]) dojdeme k obecné kategorizaci testů, kde základní členění vypadá takto:

A. **Funkční testy** - ověřují funkční požadavky na systém. Funkční požadavky definují, CO by systém měl vykonávat, např. uchovávat historii stavů objednávek v čase.

B. **Výkonnostní testy** - ověřují výkonnostní požadavky na systém. Výkonnostní požadavky definují metriky systému při specifikovaných podmínkách, např. počet zpracovaných objednávek za jednotku času při provozním zatížení systémů.
C. **Ostatní testy** - zahrnují testy, které nelze zařadit do předchozích skupin. Příkladem je např. kontrola dodané dokumentace.

U **funkčních** testů dále rozlišujeme tyto podskupiny:

A1. Unit testy(někdy též komponentové) - ověřují funkčnost samostatných komponent systému.
A2. Integrační testy - ověřují spolupráci komponent systému mezi sebou.
A3. Systémové testy - ověřují spolupráci komponent systému s externími komponentami z jiných systémů, tedy spolupráci nového systému s okolními systémy.
A4. Migrační testy - ověřují způsob plnění systému daty z jiných (např. nahrazovaného) systémů, zpravidla při úvodním plnění (tzv. initial load) databáze.
A5. Bezpečnostní testy - ověřují způsob zabezpečení systému.
A6. Regresní testy - ověřují, zda nový systém zachovává funkcionalitu nahrazovaného systému.

U **výkonnostních** testů rozlišujeme tyto podskupiny:

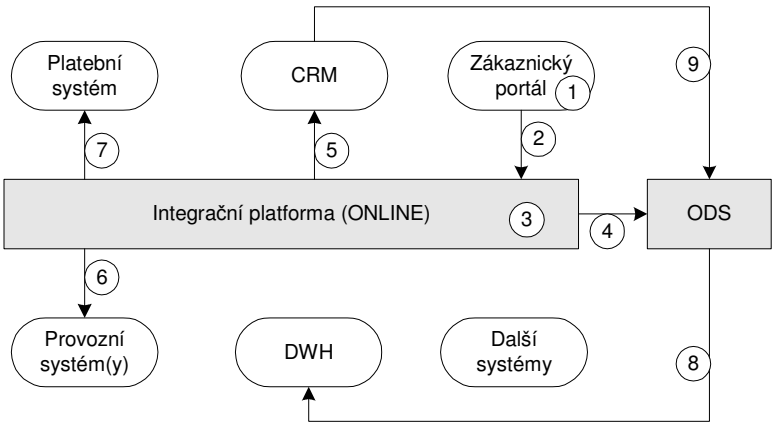
B1. Zátěžové testy - vyhodnocuje se chování systému na a za hranicemi jeho maximální zátěže, často se takto i tyto hranice hledají.
B2. Testy při pracovním zatížení (někdy též load testy) - vyhodnocuje se chování systémů za běžných (simulovaných) provozních podmínek.

V kategorii **ostatních** testů se často setkáváme s:

C1. Instalačními testy - kontrolují kompletnost dodávky a postup instalace.
C2. Testy dokumentace - kontrolují kompletnost dokumentace.
C3. Testy zotavení - kontrolují schopnost systému se zotavit do provozního stavu v případě nenadálé události, např. selhání systému.

V praxi se setkáváme i s jiným členěním testů, často se i částečně liší význam testu, ačkoli má shodný název. Uvedené kategorie nám tvoří obsahový rámec libovolného testování softwarového produktu a to nám pro naše další zkoumání postačuje.

15.2 Model integrovaného PIS



obrázek 53 - Model integrovaného PIS (zdroj: autor)

V této kapitole si navrhneme model integrovaného PIS, na kterém si v následujícím textu popíšeme základní testovací schéma. Cílem námi modelovaného integračního projektu je nahradit stávající integraci mezi 5 systémy, kterými jsou platební systém, CRM, zákaznický portál, provozní systém, datawarehouse (DWH), novou integrační platformou. Pilotním procesem, který máme podporovat, je zpracování nové objednávky ze zákaznického portálu. Na nový systém jsou kladeny 3 základní funkční požadavky:

1. Zpracování objednávek v reálném čase.
2. Dávkové předání objednávek do datawarehousu jednou měsíčně.
3. Záložní plnění ODS z CRM, které bude využito i pro úvodní naplnění ODS.

Takto formulované zadání obsahově odpovídá průměrně složitému integračnímu projektu (v modelu dochází ke zjednodušení v počtu podporovaných procesů) a můžeme jej tedy použít pro naši analýzu integračního testování.

Použití integrační platformy prezentuje topologii hub nebo bus (nebudeme dále upřesňovat, neboť z pohledu testování to není podstatné). Na ODS můžeme nahlížet jako na datový hub, ale to není z pohledu testování rovněž podstatné. Point-to-point topologie se nám objevuje mezi CRM a ODS a mezi ODS a DWH, pokud chápeme ODS jako samostatný systém a nikoli jako zprostředkující systém.

Pokud bychom museli testovat integrovaný systém v topologii point-to-point, tak to znamená připravit testování každého spojení zcela specificky dle užitých technologií spojení. V případě zprostředkovatele ve formě integrační platformy si vytvoříme pouze jednu formu testování, kterou obměňujeme podle testovaných integračních případů (bude upřesněno dále).

V modelu navrhujeme použití funkčních rozhraní pro komunikaci s integrační platformou a použití datových rozhraní pro přenos dat z CRM do ODS a z ODS do DWH. Integraci pomocí prezentačních rozhraní nemáme v modelu zastoupenou, neboť její využití je atypické.

Testování integrace na datových rozhraních je významně jednodušší, neboť se zabýváme pouze samotnými daty. V případě integrace na funkčních rozhraních se musíme nad to zbývat i obchodní logikou, která s daty pracuje, což je potenciální zdroj dalších problémů. Tyto integrace se liší i technologicky, což znamená i nutnost použít odlišné postupy pro jejich testování.

V našem modelovém příkladu nalezneme tři úrovně integrace. Typickou datovou integraci představuje např. předávání dat z ODS do DWH. Ale můžeme ji nalézt i při přenosu informací o objednávce mezi zákaznickým portálem a CRM. Integraci služeb potenciálně objevíme např. při požadavku na realizaci objednávky z CRM na provozní systém. To může být realizováno buď přímo voláním konkrétní procedury v provozním systému nebo voláním služby na realizaci objednávky, kde bude služba nalezena pomocí prostředníka (SB). Procesní integraci reprezentuje celý proces zpracování objednávky.

Datová integrace je pro testování opět relativně nejjednodušší, neboť se jedná pouze o předání dat. S integrací služeb složitost vzrůstá, jelikož přibývá interní logika služeb, u služeb navíc i požadavek na jejich obecnou použitelnost. Procesní integrace vyžaduje testování řady sekvencí celého procesu včetně transakčního zpracování.

Námi zvolený modelový integrační projekt je dostatečně komplexní (řeší integrační úlohu nahrazení integrační platformy), abychom si na něm ukázali vše potřebné pro testování. Podívejme se nyní na model detailně. V prvé řadě je potřeba identifikovat všechny nahrazované (resp. podporované) integrační procesy. Tyto procesy se nám rozpadnou na integrační případy.

Jako **vzorový příklad** jsem zvolil integrační proces zpracování objednávky po jejím přijetí přes zákaznický portál. Posloupnost jednotlivých činností procesu je zachycena očíslováním v obrázku.

1. Přijetí objednávky v zákaznickém portálu
2. Předání integrační platformě
3. Zpracování v integrační platformě
4. Předání do ODS
5. Předání do CRM
6. Realizace (nastavení) služby v provozních systémech
7. Nastavení platby za službu v platebním systému

Tento proces se nám rozpadá na několik integračních případů:

1. Předání objednávky z portálu do ODS a CRM (iniciováno objednávkou v zákaznickém portálu) (2 případy).
2. Realizace objednávky z CRM do provozních systémů (iniciováno potvrzením objednávky v CRM).
3. Platba za službu v platebním systému (iniciováno potvrzením z provozních systémů).

Řízení objednávkového procesu přebírá integrační platforma, realizuje se v reálném čase (online). Integrace probíhá na aplikační vrstvě aplikací. Aplikace poskytují rozhraní formou služeb nebo procedur, úroveň integrace je procesní.

Druhý zvolený **vzorový příklad** popisuje předání dávky dat o objednávkách z ODS do DWH, které probíhá jednou měsíčně (8). Identifikujeme jeden integrační případ, komunikaci z ODS do DWH. Je realizován nezávisle na integrační platformě formou databázového skriptu (při složitější integraci se využívá nástrojů Extract Transform Load [NOVOTNÝ, 2005]) na datové vrstvě aplikací. Úroveň integrace je datová.

Třetí vzorový příklad je určen pro počáteční nahrání dat do ODS (9), kde primárním zdrojem je CRM. Tato cesta slouží také jako záloha v případě poškození ODS pro její obnovení (v dalším textu ji budeme označovat také jako dávkovou (bulk)). Tento integrační případ se realizuje také formou databázového skriptu. Realizuje se ad hoc, na datové vrstvě. Úroveň integrace je opět datová.

Nyní se podívejme, jaké integrační komponenty jsou v modelovém řešení užity. Jsou to tyto prvky:

1. Integrační server (online), pomocí které jsou monitorovány a řízeny real-timeové integrační procesy (viz kapitola 9 Integrační proces a integrační případ).
2. Aplikační a technologické adaptéry - umožňují definovat rozhraní online připojených systémů, tedy konkrétně zákaznického portálu, CRM, provozních systémů a platebního systému, pro komunikaci s integrační platformou.
3. Operational Data Store je obecně centrálním integračním prvkem na datové úrovni, přestože v našem modelu poskytuje data pouze pro DWH.
4. Databázová rozhraní pro komunikaci mezi CRM a ODS a mezi ODS a DWH.

15.3 Testování integrovaného PIS

Nyní stojíme před úkolem otestovat nově navržený integrovaný systém, zda splňuje požadavky na něj kladené. Jaké testy budeme provádět? Vyjdeme z kategorizace testů v kapitole 15.1 Typologie testů.

Typ testu	Testovaná oblast	Korektní chování
Unit testy	Adaptér zákaznického portálu	Zachytává veškeré objednávky a předává je ve správném formátu prostřednictvím API.

	Adaptér CRM	Zachytává příchozí objednávky prostřednictvím API, předává je ke zpracování v CRM a vrací přes API odpověď o zpracování.
	Adaptér provozních systémů	Zachytává příchozí objednávky prostřednictvím API, předává je ke zpracování v provozním systému a vrací přes API odpověď o zpracování.
	Adaptér platebního systému	Zachytává příchozí objednávky prostřednictvím API, předává je ke zpracování v platebním systému a vrací přes API odpověď o zpracování.
	Adaptér ODS	Zachytává příchozí objednávky prostřednictvím API, předává je ke zpracování v ODS a vrací přes API odpověď o zpracování.
	Proces zpracování objednávky, 1. část	Zachytává příchozí objednávky prostřednictvím API, provádí transformaci/zpracování zprávy do formátů ODS a CRM a předává je ke zpracování do ODS a CRM přes API.
	Proces zpracování objednávky, 2. část	Zachytává potvrzené objednávky prostřednictvím API, provádí transformaci/zpracování zprávy do formátu provozního systému a předává je ke zpracování do provozního systému přes API.
	Proces zpracování objednávky, 3. část	Zachytává potvrzené realizované objednávky prostřednictvím API, provádí transformaci/zpracování zprávy do formátu platebního systému a předává je ke zpracování do platebního systému přes API.
	Struktura ODS	Struktura a chování (např. triggery a integritní pravidla) ODS odpovídá požadavkům.
	Datové API CRM, read	Poskytuje vybraná data o objednávkách z CRM databáze v definované struktuře a formátu.
	Datové API ODS, write	Načítá data o objednávkách z datového API CRM, transformuje je, zpracovává a zapisuje (dle nastavených business pravidel INSERT nebo MERGE) do struktur ODS.
	Datové API ODS, read	Poskytuje vybraná data o objednávkách z ODS databáze v definované struktuře a formátu.
	Datové API DWH, write	Načítá data o objednávkách z datového API ODS, transformuje je, zpracovává a zapisuje (dle nastavených business pravidel INSERT nebo MERGE) do struktur DWH.
Integrační testy	Proces zpracování objednávky, 1. část	Zpracování přijaté objednávky z portálu a její předání do ODS a CRM s využitím příslušných adaptérů, které však nemusí být ještě připojeny k příslušným systémům (tzv. fake).

	Proces zpracování objednávky, 2. část	Zpracování potvrzené objednávky z CRM a její předání do provozních systémů s využitím příslušných adaptérů, které však nemusí být ještě připojeny k příslušným systémům (tzv. fake).
	Proces zpracování objednávky, 3. část	Zpracování realizované objednávky z provozních systémů a její předání do platebního systému s využitím příslušných adaptérů, které však nemusí být ještě připojeny k příslušným systémům (tzv. fake).
	Proces zpracování objednávky, kompletní	Posloupnost předchozích testů.
	Initial load z CRM do ODS, insert	Nahrání modelových dat z CRM databáze do prázdné ODS.
	Backup load z CRM do ODS, merge	Nahrání modelových dat z CRM databáze do poloprázdné ODS.
	Předání dat z ODS do DWH	Předání modelových dat z ODS do DWH.
	Komparace transformace objednávek do ODS online a dávkovým způsobem	Transformační proces v online probíhá shodně s dávkovým, tedy výsledek postupného (online) zadávání objednávek se v ODS jeví shodně s výsledkem nahrání stejných objednávek přes dávkové zpracování z CRM.
	Administrace procesu	Umožňuje řídit proces manuálně (zasahovat do probíhajícího zpracování, např. restartovat probíhající proces, řešit chybové stavy).
	Monitorování procesu	Integrační platforma poskytuje informace o stavu procesu.
	Administrace integrační platformy	Umožňuje administrovat komponenty systému (přístupy apod.).
	Monitorování integrační platformy	Integrační platforma poskytuje informace o stavu jednotlivých komponent a upozorňuje na kritické situace.
Systémové testy	Shodné s integračními testy, avšak již na produkčních datech a systémech (míněno v testovacím prostředí).	
Migrační test	Zde realizován již formou integračního testu Initial load z CRM do ODS, insert.	
Bezpečnostní testy	Adaptér zákaznického portálu, CRM, provozního systému a platebního systému	Odmítnutí neautorizovaného volání adaptéru.
	Proces zpracování objednávky, komunikace	Nelze odchytit (číst nebo smazat), pozměnit, podstrčit předávané zprávy (např. šifrovaná komunikace).

	Proces zpracování objednávky, správa	Odmítnutí neautorizovaného přístupu k řízení procesu.
	ODS	Nelze narušit integritu dat ani pozměnit data jinak než přes datové API ODS.
	Datová API ODS, CRM, DWH	Odmítnutí neautorizovaného přístupu k datům.
	Předání dat z CRM do ODS a ODS do DWH, komunikace	Nelze odchytit (číst nebo smazat), pozměnit, podstrčit předávaná data (např. šifrovaná komunikace).
Regresní testy	Proces zpracování objednávky, kompletní	Proces probíhá shodně se způsobem zpracování objednávky v předchozím systému (ve vztahu k integrovaným systémům i způsobu řízení objednávky), nalezené rozdíly jsou v souladu s navrženými změnami procesu při přechodu na nový systém. V tomto testu bychom měli plně využít i integrované systémy, aby proces probíhal od začátku až do konce.
	Předání dat z ODS do DWH	Data jsou do DWH dodávána shodně s předchozím systémem, nalezené rozdíly jsou v souladu s navrženými změnami procesu při přechodu na nový systém.
Zátěžové testy	Maximální počet zpracovaných objednávek procesem	Zjišťuje se, kolik objednávek je schopen integrovaný systém zpracovat za běžných provozních podmínek (simulovaná zátěž na CRM, zákaznickém portálu, provozních systémech, platebním systému) postupně zvyšovaným počtem objednávek.
	Hledání úzkého místa (tzv. bottlenecku)	Zvyšování zátěže jednoho ze systémů při provozní zátěži zbývajících systémů. Cílem je nalezení kritické hodnoty.
Testy při pracovním zatížení	Průměrný počet zpracovaných objednávek	Průměrný počet zpracovaných objednávek za běžných pracovních podmínek.
	Maximální odezva systému při zpracování objednávky	Jak dlouho trvá zpracování objednávky v jednotlivých krocích za běžných pracovních podmínek.
	Zátěž CRM za běhu dávkového procesu	Kombinace běhu dávkového zpracování a provozní zátěže CRM, dopad na CRM.
	Zátěž DWH za běhu nahrávání dat z ODS	Kombinace běhu dávkového zpracování a provozní zátěže DWH, dopad na CRM.
Instalační testy	Kompletnost dodávky	Dodávka integračního řešení obsahuje všechny komponenty a náležitosti.
	Instalace integračních komponent	Instalace proběhla bez závad.
	Odinstalace	Odinstalace proběhla bez závad.
Test dokumentace	Kompletnost a úroveň dokumentace	Dokumentace je kompletní a obsahuje instalační příručku, technickou specifikaci a provozní dokumentaci na potřebné úrovni.
Test zotavení	Obnovení ODS po zhroucení	ODS naběhne bez závad a iniciuje proces doplnění chybějících dat během výpadku.

	Obnovení integrační platformy po zhroucení	Integrační platforma naběhne bez závad, obnoví probíhající procesy zpracování objednávek a zpracuje i objednávky přišlé během výpadku.
	Nedostupnost zákaznického portálu, CRM, provozního systému a platebního systému při online zpracování	Proces zpracování objednávek detekuje nedostupnost systému a po jeho obnovení pokračuje dál ve zpracování.
	Nedostupnost CRM, DWH při dávkovém zpracování	Dávkový proces detekuje nedostupnost systému a po jeho obnovení pokračuje dál ve zpracování.

tabulka 20 - Typické příklady testů v integračním projektu (zdroj: autor)

Několik poznámek k navrženým testům:

1. Všimněme si, že unit testy a především integrační testy odpovídají identifikovaným integračním případům.
2. Tento přehled testů považujme pouze za základní rámec testů, nezabývali jsme se například vůbec testováním chybových stavů.
3. Rovněž jsme nenavrhovali testy pro ověření integrační platformy. Její funkcionalita je zpravidla garantována jejím dodavatelem a její posouzení je součástí výběrového řízení.
4. Předpokládá se, že testování probíhá na samostatném testovacím prostředí, které musí obsahovat nejen integrační komponenty, ale i testovací verze připojovaných systémů.
5. Pro výkonnostní testování je zpravidla třeba vymyslet způsob, jak testy provést v produkčním prostředí, aby výsledné hodnoty odpovídali realitě. Alternativně mají testovací týmy určenu empirickou hodnotu vztahu výkonnosti mezi testovacím a produkčním prostředím, kterou lze využít pro přepočet hodnot z testovacího prostředí. V takovém případě je ale nutno počítat s vyšší chybou a rizikem.
6. Velmi důležitým testem je "Komparace transformace objednávek do ODS online a dávkovým způsobem". Jelikož je dávkové zpracování objednávek z CRM do ODS záložním způsobem plnění ODS, musí obsahovat stejnou business logiku jako online zpracování objednávek. V realitě to znamená vyvíjet identický kód na různých technologiích a tím zde dochází nejen k duplicitě nákladů na vývoj, ale i potenciálním rozdílům (chybám) plnění ODS.

15.4 Charakteristika integračního testování

Na příkladu jsme si ukázali typické testování nového integrovaného podnikového informačního systému. Můžeme konstatovat, že představuje komplexní soubor testů ve všech kategoriích, které se aplikují na jiné typy softwarových produktů. Nalezli jsme i několik odlišností :

1. Unit testy se zaměřují na testování adaptérů a částí integračních procesů.
2. Integrační testy vycházejí z integračních případů, což jsou analyticko-designové prvky specifické pro integrační projekty.
3. Testování je náročné na přípravu testovacího prostředí, neboť musí obsahovat i testovací verze integrovaných systémů.
4. Pro kritické výkonnostní testy je třeba navrhnout způsob provedení testů na produkčním prostředí.
5. V případě vytváření záložní cesty (v našem případě předání dat z CRM do ODS) k online zpracování je nutné důkladně otestovat, zda oba procesy poskytují identické výstupy.

6. Oproti standardním aplikacím se netestují uživatelská rozhraní (GUI).
7. Často se užívají porovnávací testy, neboť nahrazovaný systém nějakým způsobem integrační proces již realizoval, a je tedy možné provést komparaci jeho výstupů s výstupy nového integračního procesu.
8. V praxi se klade velký důraz na výkonnostní testy a testy zotavení, aby integrační platforma nebyla úzkým místem integrovaného PIS. Tento důraz je dán tím, že integrační platforma je zpravidla považována za kritickou aplikaci (její výpadek by měl rozsáhlé negativní dopady na fungování podniku).

15.5 Obecný proces testování

V této části se budeme věnovat výhradně fázi testování a aktivitám, které s ní souvisí. Popíšeme si celý proces přípravy testování a průběh testování. Tento proces je popsán v dostupných informačních zdrojích a zde si jej popíšeme pouze pro sjednocení terminologie a ujasnění základní linie testování a jeho souvislostí. Smyslem není duplikovat tyto informace, ale rozpracovat tento proces na klasický integrační projekt.

Hlavní část našeho zkoumání bude směřovat k testovací fázi v integračních projektech. Nebudeme se zabývat obsahovou částí testování, ale především jeho organizační složkou. Pokusíme se identifikovat odlišnosti při přípravě, organizování a řízení testování v případě čistě integračních projektů.

Budeme postupovat podle následující logické linie. Nejdříve si představíme obecný proces testování tak, jak jej definují dostupné zdroje, včetně vstupů, výstupů a hlavních činností. V druhém kroku si v hrubých rysech vymezíme klasický integrační projekt, abychom jej mohli analyzovat při hledání jeho specifických vlastností ve fázi přípravy a vlastního testování.

Čtenář tak získá konkrétní představu, jak v optimálním případě probíhá testování v komplexním integračním projektu, a na co je třeba se soustředit po organizační stránce. Tyto informace jsou užitečné zejména v případě, že nemá s testováním žádné zkušenosti nebo má zkušenosti s testováním jiné kategorie softwarových produktů.

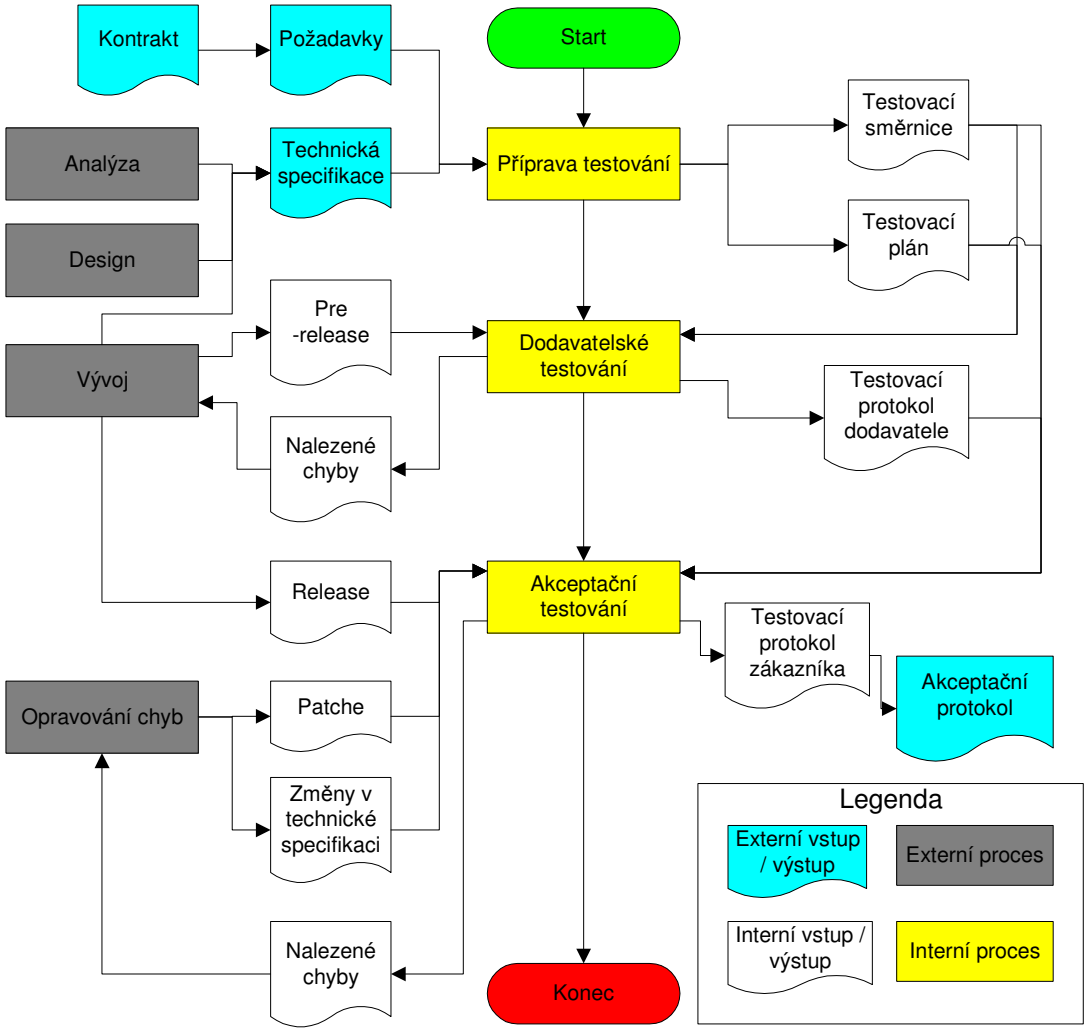
Obrázek 54 - Schéma testovacího procesu (zdroj: autor) znázorňuje základní závislosti testovacího procesu. Celý proces dělíme do tří částí :

1. Příprava testování, která probíhá souběžně s fází analýzy, designu a vývoje. Jejím cílem je naplánovat a připravit vše potřebné pro hladký běh testování.
2. Dodavatelské testování obsahuje testování prováděné dodavatelem projektu. Začíná v průběhu vývoje tzv. unit testy a pokračuje testy, které jsou shodné nebo obdobné s akceptačními testy.
3. Akceptační testování představuje závěrečné testování na straně odběratele (dále též zákazník), zda dodaný produkt odpovídá požadavkům ze smlouvy.

Takto popsaný proces prezentuje tradiční přístup k testovací fázi a budeme se jej držet i nadále. V informačních zdrojích ([BECK, 2002], [HOHPE, 2002b], [CRISPIN, 2002], [FOWLER], [GOLD, 2004]) se setkáváme i s odlišným přístupem (metodika extrémního programování), který využívá připravených testovacích příkladů jako jeden ze vstupů při vývoji produktu. Další vývojové přístupy pracují s postupným předáváním komponent k dodavatelskému testování, nikoli s kompletní dodávkou v určený čas (viz [HOHMANN, 2003]). Tyto přístupy jsou alternativní cestou pro vývoj produktu. Jelikož však nemají dopad na způsob testování, nebudeme je dále zkoumat.

Pro pochopení konfliktu mezi vývojem a testováním je třeba si uvědomit, že primárním cílem testování je vyhodnotit stav dodaného produktu (viz [SOPHOS, 2005]) (tedy nalézt co nejvíce chyb) a naopak cílem vývoje je dodat funkční produkt v požadovaném termínu (tedy minimalizovat počet chyb). Z toho je třeba vycházet při plánování testů - proces testování je zde nezávislým procesem na procesu vývoje (i opravování chyb) a

skončí hodnocením produktu, nikoli až když jsou opraveny všechny chyby. Jelikož cílem projektu je ale dodat bezvadný produkt plánují se uměle tzv. **testovací cykly**, které periodicky provedou vyhodnocení produktu a identifikují chyby. Kvalita dodávaného produktu tak ovlivňuje pouze počet potřebných (nikoli však nutně realizovaných) testovacích cyklů. Podrobněji se tomuto tématu věnuje [MCGREGOR, 2001].



obrázek 54 - Schéma testovacího procesu (zdroj: autor)

15.5.1 Příprava testování

Smyslem přípravy testování je nastavit pravidla v testovací směrnici pro běh testování a v testovacím plánu časově a organizačně naplánovat běh testovacích příkladů. My se budeme primárně zabývat sestavením těchto dokumentů pro akceptační testování, ačkoli mohou vzniknout i pro dodavatelské testování (záleží na metodice dodavatele).

Testovací směrnice (též akceptační plán, někdy je součástí testovacího plánu) stanoví pravidla pro spolupráci mezi dodavatelem a zákazníkem během testování (můžeme to chápat jako část integrace metodik zákazníka a dodavatele). V ideálním případě jsou tato pravidla stanovena přílohou ke smlouvě. Pokryty by měly být tyto oblasti:

1. Předpoklady testování.
2. Jednotná definice pojmů (testovací příklad, testovací scénář, testovací plán, testovací běh, akceptační protokol), kategorií testů a další vymezení testování.
3. Rozsah a granularita testování (např. definice funkčních oblastí).

4. Řízení změn (detailněji viz [HASS, 2002]) pro oblast testovacích příkladů, releasů a dokumentace, zákaznických požadavků.
5. Řízení chyb (popis chyby včetně identifikace, závažnosti, klasifikace a vlastníků; chybové workflow).
6. Řízení testovacích běhů (definice test plánu, šablona pro testovací příklady (vzor viz IEEE standard 829-1983 [HORCH, 2003]), workflow testovacích příkladů, klasifikace testovacích příkladů).
7. Eskalační procedura pro řešení konfliktů.
8. Sestavení testovacího týmu s přidělením pravomocí a odpovědností (správa testovacích prostředí, reportování, řízení testů, eskalační procedury, funkční oblasti) (viz [SOPHOS, 2005] a [DUSTIN, 2002]).
9. Specifikace testovacích prostředí.
10. Testovací nástroje (o typech testovacích nástrojů pojednává [DUSTIN, 2002]).
11. Trackovací nástroje pro evidenci a reportování běhu testovacích příkladů a nalezených chyb.

Testovací plán definuje jednotlivé testovací příklady na detailní úrovni a organizuje je do testovacích cyklů. Šablonu testovacího plánu nalezneme v IEEE standardu 829, nebo komerčně např. v [ReadySET]. **Testovací cyklus** je plán jednoho běhu sady všech testovacích příkladů v čase. Je definován začátkem běhu testování a končí opravením nalezených chyb (viz [DUSTIN, 2002]).

Při tvorbě testovacích příkladů vycházíme z požadavků definovaných ve smlouvě mezi zákazníkem a dodavatelem, kde by u každého požadavku mělo být jasně definované akceptační kritérium. Pokud tomu tak není, tak je třeba jej doplnit během analýzy. Existují přesné návody, jak postupovat při sestavování požadavků na systém a jak z nich tvořit testovací příklady, viz např. [DUSTIN, 2002]. O požadavcích na systém v integračním projektu píši v kapitole 8 Požadavky v integračních projektech.

Druhým významným zdrojem pro definování testovacích příkladů je **technická specifikace** dodávaného systému, která vzniká během analýzy a designu a postupně se upravuje i během vývoje a opravování nalezených chyb. Technická specifikace říká, jak budou realizovány požadavky na systém.

Existuje několik metod, jakým způsobem definovat testovací příklady, podrobně se jim věnuje [SOPHOS, 2005]. Techniky se rozlišují podle otevřenosti systému na (dle [MCGREGOR, 2001], [DUSTIN, 2002], viz také kapitola 2.1 Obecná charakteristika integračních projektů):

1. Black box - testování pouze prostřednictvím externích rozhraní celého systému.
2. Gray box - testování prostřednictvím grafického rozhraní nebo přímo testováním jednotlivých komponent (sem zahrnuji i legacy aplikace).
3. White box (též strukturální, nebo implementačně orientované) - testování na základě programového kódu a vnitřní struktury systému.

Testovací příklady (test cases) jsou definovány v dokumentech s jednotnou šablonou, zahrnující identifikaci testovacího příkladu, počáteční podmínky, jednotlivé kroky a jejich očekávané výsledky, potřebná testovací data, testovací nástroje, testovací prostředí. Testovací příklady se ukládají do katalogu (inventory), o jehož správě a využití se píše více v [HUTCHESON, 2003].

Příprava testovacího plánu začíná **analýzou rizik**, jejímž účelem je identifikovat vysoce rizikové aplikační komponenty, které musí být testovány opravdu důkladně. Používají se čtyři hlavní metody analýzy rizik, pro základní orientaci jsou popsány např. v [SOPHOS, 2005]. Výstupem analýzy jsou hlavní cíle testování, resp. základní funkcionalita systému.

Při sestavování **testovacího plánu** zohledňujeme tato omezení:

1. požadované termíny předání systému do provozu,
2. dostupnost a schopnosti potřebných lidských zdrojů,
3. nutnost sdílení testovacích prostředí a nástrojů

a pracujeme s určitou předpokládanou úrovní

1. chybovosti dodaného produktu k testování a
2. schopností dodavatele opravovat chyby v reálném čase.

Na základě těchto faktorů stanovíme tzv. **pokrytí** funkcionality testováním (poměr provedených testů k sumě všech testů potřebných k maximálnímu možnému otestování produktu) [MCGREGOR, 2001].

V sadě prováděných testů identifikujeme **prioritní testovací příklady** (chyby v těchto testech budou opravovány přednostně, aby nedošlo ke zdržení během testování) dle

- závislosti mezi testovacími příklady (některé testy jsou podmíněny bezvadným během jiného testovacího příkladu),
- nadprůměrné pracnosti testovacích příkladů,
- příslušnosti k hlavním testům funkcionality systému (z analýzy rizik).

Z těchto testů sestavíme základní linii testovacího cyklu. Tu následně doplníme zbývajícími testy, čímž nastavíme i **styl testování**:

A. do hloubky (nejprve se snažíme otestovat vybranou funkční oblast),
B. do šířky (nejprve se snažíme provést průřezové testy přes všechny komponenty).

V závěru stanovíme počet testovacích cyklů, opět s ohledem na časové termíny projektu a dle předpokládané chybovosti dodaného produktu k testování. Metodami pro plánováním a odhadováním pracnosti testování se detailně zabývá [JALOTE, 2002] a [DUSTIN, 2002].

Testování probíhá různými typy testů, jejich popis je nad rámec tohoto článku (zabývá se jimi např. [FRANK, 2002], [HOHPE, 2002b], [HORCH, 2003]). Na obrázku 55 - W Model (zdroj: [HASS, 2002]) je jeden z fundamentálních modelů posloupnosti testů, tzv. W-model.

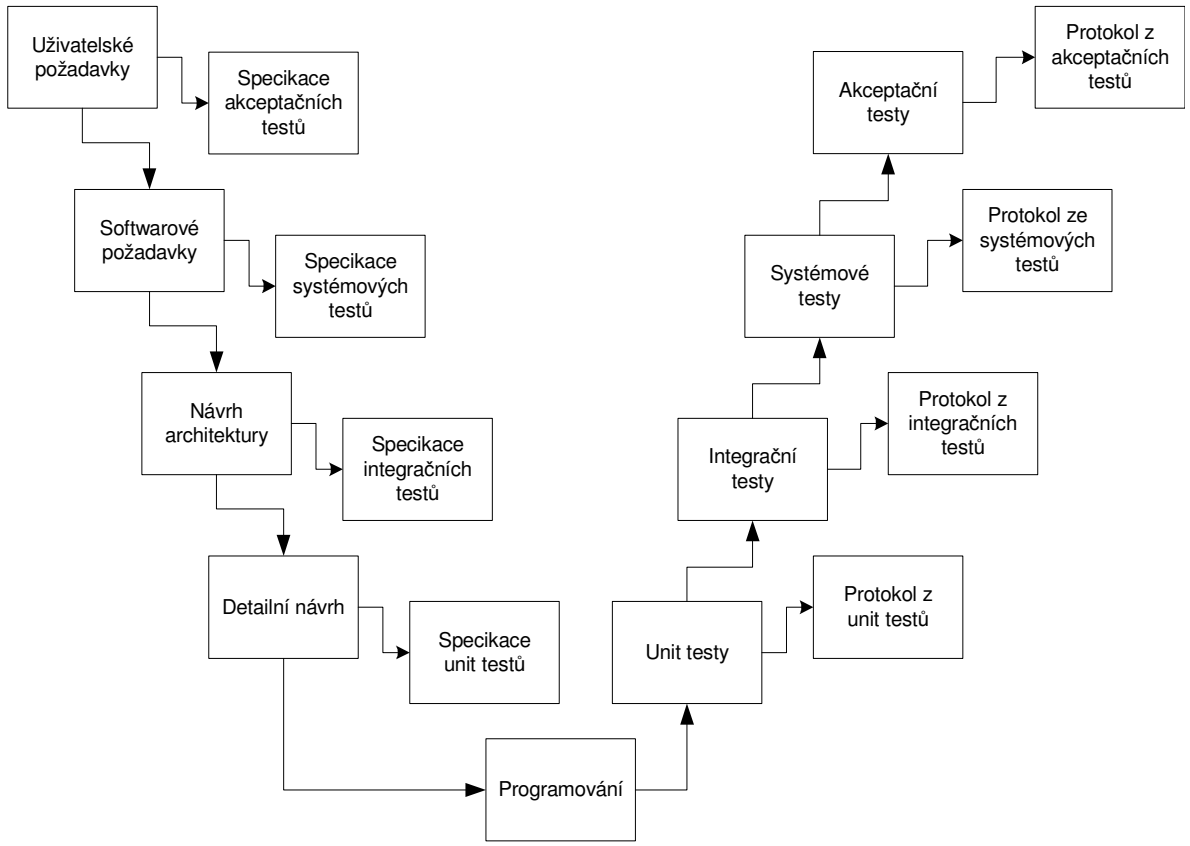
Dle testovacího plánu si alokujeme zdroje pro dobu testování, připravujeme testovací data a nástroje. Sada testovacích příkladů by měla být vzájemně odsouhlasena zákazníkem i dodavatelem již v této fázi, neboť ve fázi testování by to vedlo k časovým prodlevám při řešení konfliktů.

Za přípravu testovacího plánu zodpovídá manažer testů a připravuje jej ve spolupráci z vedoucími testovacích týmů. Za přípravu testovacích příkladů zodpovídají vedoucí testovacích týmů ve spolupráci s členy týmů.

15.5.2 Dodavatelské testování

Dodavatel provádí své interní testy (tzv. alfa testování) před předáním systému k akceptačnímu testování zákazníkovi. Tyto testy provádí ve fázi vývoje (ale může být formálně vymezena i samostatná fáze dodavatelského testování). Obsahem tohoto testování jsou tzv. unit testy (ověřují funkčnost samostatných komponent systému - dle IEEE standardu 610.12-1990 je to komponenta, která již nemůže být dále rozdělena další komponenty [HORCH, 2003]) a zpravidla se provádí i integrační testy (ověřují spolupráci komponent systému mezi sebou). Většina dodavatelů provádí i testy v souladu s akceptačními testy (tzv. beta testování), čímž se pak akceptační testování zákazníka stává spíše formální činností.

Výsledkem této aktivity je protokol o provedených testech předávaný zákazníkovi.



obrázek 55 - W Model (zdroj: [HASS, 2002])

15.5.3 Akceptační testování

Běh akceptačních testovacích příkladů probíhá dle testovacího plánů a operativně se řeší časové odchylky od plánu. Průběžně se monitoruje

1. množství a závažnost nalezených chyb (o doporučených metrikách se dočteme např. v [SOPHOS, 2005] a [KAN, 2002]) a
2. rychlost a kvalita opravování chyb

a hodnoty se porovnávají s předpoklady testovacího plánu. Úpravy testovacího plánu probíhají na konci každého testovacího cyklu.

Jak již bylo zmíněno na počátku, primárním výsledkem testování je testovací protokol, nikoli bezvadný produkt. Je na zvážení projektového manažera, zda se v případě vadného produktu testovací fáze prodlouží o několik testovacích cyklů, aby se dosáhlo požadovaného stavu produktu.

Testovací protokol popisuje průběh a výsledky běhu testovacích příkladů v jednotlivých cyklech. Nejdůležitější jsou závěry konečného běhu, neboť vypovídají o aktuálním stavu produktu. Na základě testovacího protokolu vzniká **akceptační protokol**, který vypovídá o splnění smluvních požadavků mezi zákazníkem a dodavatelem. Akceptační protokol klasifikuje produkt jako

1. akceptovaný bez výhrad - nebyly nalezeny závady, nebo
2. akceptovaný s výhradou - byly nalezeny přiložené závady, které nebrání provozu systému (a dodavatel se je zavazuje do dané lhůty odstranit), nebo
3. neakceptovaný - systém nelze provozovat vzhledem k přiloženému seznamu závad.

15.6 Proces testování v integračním projektu

Za vzorový příklad si vezmeme model z kapitoly 15.2 Model integrovaného PIS. Pro naše potřeby si tedy představme projekt, v kterém budeme implementovat novu integrační platformu pro online podporu integračních procesů mezi několika aplikacemi. V souvislosti s tím provedeme revizi konektorů jednotlivých aplikací a redefinuje integrační procesy pro novou platformu.

Vedle integrační platformy budeme implementovat také ODS (Operation Data Store), který využijeme jako zdroj dat pro integrační procesy a zároveň jako alternativní zdroj dat pro dávkovou integraci vybraných databází.

15.6.1 Požadavky

Funkční požadavky na integrační projekt zpravidla využívají jako referenční základ stávající integraci mezi systémy. Nefunkční požadavky směřují na propustnost integrační platformy a na dobu odezvy ODS. Nezřídka bývají součástí zadání i požadavky na využití průmyslových standardů v architektuře řešení (např. [REILLY, 2004], [TMF, 2003] pro ODS). Takové požadavky jsou fakticky akceptovány již schválením detailního návrhu řešení, neboť nemá smysl je testováním zkoumat. Podrobně se problematice požadavků v integračních projektech věnuji v kapitole 8 Požadavky v integračních projektech.

15.6.2 Technická specifikace

Funkcionalita integrační platformy je detailně popsána integračními procesy s rozpadem na integrační případy. V návaznosti na ně jsou popsána rozhraní integrovaných aplikací (nutné úpravy v těchto aplikacích tvoří samostatné zadávací dokumenty pro správce aplikací). ODS je popsán ve formě datového modelu a poskytovaného rozhraní (pro online i dávkovou integraci). Integrační případy dávkové integrace jsou popsány samostatně dle aplikací.

Sadu testovacích příkladů odvozujeme z požadavků. Funkční testovací příklady strukturujeme podle integračních případů (procesů) (v syntaxi UML vycházíme z use case modelů dle [KRUCHTEN, 2003]) popsaných v technické specifikaci.

15.6.3 Testovací tým

Při integračním testování můžeme zachovat standardní strukturu testovacího týmu, který vede testovací manažer s pomocí testovacích lídrů podle funkčních oblastí. Na úrovni testerů bude tvořit heterogenní skupinu, neboť musí být využiti specialisté na jednotlivé integrované systémy, aby bylo možno otestovat i dopad nové integrace na okolní systémy. Jádro týmu tvoří experti na integrační platformu, což při nasazování nového integračního produktu požaduje jejich důkladné zaškolení. Třetí větev týmu představují databázoví specialisté, kteří budou testovat ODS a dávkovou integraci. Čtvrtá skupina bude tvořena testery na nefunkční požadavky, tj. výkonnost, dostupnost, zabezpečení, atd. Rozsah testovacího týmu se pro středně velký projekt pohybuje v desítkách testerů.

Partnerem pro testovací tým je tým vývojářů, kteří budou opravovat nalezené chyby. Pro efektivní komunikaci se doporučuje struktura vývojářského týmu zrcadlově obdobná s testovacím týmem.

15.6.4 Testovací prostředí

Integrační projekty mají velké nároky na testovací prostředí. Testovací prostředí musí zahrnovat nejen integrační platformu a databázi pro ODS, ale i plně funkční testovací verze integrovaných aplikací. Tyto testovací verze by měly být připraveny včetně úprav požadovaných integračním projektem.

V realitě některé systémy svůj testovací klon k dispozici nemají, nebo je plně vytížen jinými projekty. V takových případech se využívají buď **fake nástroje** (uměle vyvinuté nástroje, které simulují chování rozhraní systému). Tento přístup však významně zvyšuje riziko selhání v produkčním prostředí a lze jej akceptovat jen u méně významných aplikací.

Dalším problémem testovacích prostředí je požadavek na testování v produkčním prostředí. Cílem této snahy je aplikovat testy na produkčních datech a ověřit reálné výkonnostní parametry integrační platformy. Pro řešení tohoto požadavku asi nelze říci obecné pravidlo a postupuje se podle konkrétních možností.

Při plánování testování se vždy klade velký důraz na maximální možnou paralelnost běžících testů, aby testovací cykly probíhaly v krátkých periodách. Komplexnost integračního testování vede k tomu, že se při testování využívají až 4 prostředí současně. Jedno z nich využívá vývojový tým pro opravování chyb a ostatní jsou určena pro testovací týmy, aby se zamezilo vzájemným konfliktům mezi běžícími testy. Pro úsporu nákladů stačí při budování těchto prostředí efektivně duplikovat jen klíčové komponenty řešení (např. integrační platformu), zatímco u integrovaných aplikací bude k dispozici jen jeden testovací klon pro všechna prostředí.

Testovací prostředí se nebudují pro jednorázové využití, ale i pro další rozvoj integrační platformy. Sestavení testovacích prostředí v integračních projektech tak není jednoduchou ani levnou záležitostí.

15.6.5 Testovací nástroje

Testovací nástroje jsou prostředkem podpory a urychlení testovacího procesu. Na jejich přípravě se pracuje společně s přípravou testovacích dat již před začátkem spuštění testování. Zamysleme se nad nástroji, které bychom využili v našem modelovém případu.

Testovací nástroj by obecně měl generovat (nebo jinak získat) testovací data, řídit probíhající test (pomocí tzv. testovacího skriptu, který definuje jednotlivé kroky testu s jeho vstupy a očekávanými výstupy) a vyhodnocovat jeho výsledek formou reportu. Základní vlastností užití testovacího nástroje by měla být snadná opakovatelnost testu.

Pro testování bychom potřebovali nástroje pro:

1. testování integračních procesů na integrační platformě (automatizovaný test základní funkcionality se nazývá "kouřovým" (smoke) testem, více viz [DUSTIN, 2002]),
2. testování chování systému během regulované zátěže (integrační platformy i integrovaných systémů),
3. testování odlišností v chování rozhraní integrovaných systémů před a po nasazení nového integrovaného systému,
4. testování odlišností mezi online a dávkovými integračními procesy v případě, že oba procesy mají produkovat shodné výstupy (např. jeden je určen jako záložní nebo pro úvodní naplnění dat (tzv. initial load).

Nástroj č. 4 budeme patrně realizovat pomocí databázové procedury, kterou si sami připravíme. Nástroj č. 3 je silně závislý na typu integrované aplikace, nelze jej více konkretizovat. K simulaci zátěže a potřebným měřením (nástroj č. 2) existuje řada univerzálních komerčních nástrojů, kterými se zde nebudeme dále zabývat. Nástroj č.1 je specifický pro integrační projekty a je koncepčně popsán v kapitole 12.1.9 Testovací nástroje.

Pro integrační projekty lze doporučit maximální podporu testovacími nástroji. Důvodem je, že veškerá rozhraní jsou systémová, nikoli uživatelská, což usnadňuje nasazení automatických nástrojů. Druhým důvodem je nutnost počítat s častou žádostí o přetestování funkcionality v budoucnu, při rozvoji integrovaných aplikací. **Frekvence testování** je dána frekvencí změn ve všech integrovaných aplikacích s potenciálním

dopadem na integrační rozhraní a to vše kombinováno s frekvencí změn na integrační platformě.

Podpůrnými nástroji jsou fake nástroje, které pomáhají simulovat chování aplikací, které nemohou být přímo zapojeny v testovacím prostředí. Ty je potřeba vyvinout v potřebném předstihu, protože budou potřebné již během vývoje.

15.6.6 Testovací data

Pro testování budeme potřebovat velké množství různorodých dat. Jednotlivé integrované aplikace musí mít připravena všechna vzorová data pro integrační platformu i dostatečně velké dávky dat na testování výkonnosti platformy. Totéž platí pro dávkově integrované aplikace.

V ideálním případě jsou k dispozici umělá data (generovaná bez potenciálních chyb) i provozní data (zatížená chybami z integrovaných aplikací). S chybami v datech je nutné při integraci zásadně počítat a systém na ně musí být připraven automaticky reagovat. Pro efektivní testování je nezbytně nutné rozlišovat testy s bezvadnými a chybnými daty, aby nedocházelo ke zmatkům při analýze reportovaných chyb.

15.6.7 Testovací plán

Analýza rizik by měla identifikovat klíčové integrační procesy, základní funkcionalitu ODS a některé dávkové integrační procesy. V integračním projektu je třeba prosazovat maximální možné pokrytí funkcionality testováním. To je dáno tím, že přes integrační platformu standardně protékají statisíce zpráv denně a v případě chyby je i v případě rychlé opravy problém nastolit zpět konzistenci mezi systémy.

Integrační testování začínáme instalačním testem, pokračujeme unit testováním konektorů aplikací, integračních případů (online i dávkových), ODS (strukturou a rozhraní) a dále nahráním dat do ODS (tzv. initial load). Následně provedeme integrační testy základní funkcionality. Od tohoto bodu nastává silně paralelní testování, neboť se rozbíhá testování detailní funkcionality (které představuje minimálně několik stovek testovacích příkladů) a zároveň mohou být postupně prováděny i testy nefunkčních požadavků. Pokud úpravy integrovaných aplikací provádí třetí strana, pak je třeba zajistit návaznost dodaných úprav a otestování aplikací na dodání nové integrační platformy a její testovací plán. Celý cyklus testování by měl být zakončen systémovými testy mezi koncovými aplikacemi a pre-produkčním (v de facto produkčních podmínkách) testem v reálném prostředí.

15.6.8 Ostatní

Ve zbývajících částech procesu přípravy testování jsem nenašel významnější odlišnosti či specifické rysy pro integrační projekty. Rovněž tak i pro zbývající procesy dodavatelského a akceptačního testování, do kterých se promítá vše výše zmíněné formou testovacího plánu a směrnice. Samotný běh testů již probíhá podle obecných pravidel testování.

15.7 Procesní odlišnosti testování v integračních projektech

Shrneme-li výsledky naší analýzy testovací fáze integračních projektů, tak můžeme konstatovat, že příprava i vlastní testování probíhá shodným způsobem jako testování jiných softwarových produktů. Našli jsme několik zajímavých odlišností, na které je třeba se při plánování integračního testování soustředit:

1. Požadavky na integrovaný podnikový systém jsou zcela konkrétní ve vztahu ke stávajícímu stavu podnikového systému.
2. Technická specifikace by měla obsahovat popis integračních případů, resp. procesů, které jsou předobrazem pro testovací příklady.

3. Testovací tým je heterogenní (obsahuje i specialisty na jednotlivé integrované systémy) a je třeba vyškolit experty na integrační platformu.
4. Testovací prostředí musí obsahovat i integrované aplikace.
5. Jedno testovací prostředí je zpravidla nedostačující.
6. Existuje několik typů testovacích nástrojů, které jsou typické pro integrační testování.
7. Je zapotřebí připravit velké množství testovacích dat (uměle generovaných i produkčních).
8. Běh testování umožňuje vysoce paralelní činnost.

16. Závěr

V práci jsou dokumentovány charakteristické rysy vnitropodnikových integračních projektů, které se podařilo identifikovat analýzou dostupných informačních zdrojů a pozorováním reálných projektů.

Budeme-li postupovat podle životního cyklu projektu (v souladu s logickým členěním práce), pak se jedná o tyto hlavní charakteristiky:

- Integrační projekty jsou podřízeny formulované **integrační strategii**, která definuje pravidla užití integračních přístupů v rámci jednoho podniku. Více v kapitole 10.3 Integrační koncepce.
- Přínosy v integračních projektech jsou odvozovány od zautomatizování části podnikových procesů. Efektivnost podnikových procesů je ovšem závislá i na samotných integrovaných aplikacích, proto se přínosy integračních projektů někdy obtížně vyčíslují. Více v kapitole 7.2 Přínosy.
- Z pohledu projektového řízení je kladen velký důraz na minimalizaci rizik projektu. Ta jsou dána několika faktory, mezi které patří vysoké požadavky na kvalitu implementace integrační platformy jako **kritické podnikové aplikace** a vysoce variabilní prostředí způsobené průběžnými změnami v integrovaných aplikacích. Více v kapitole 7.4 Rizika.
- Požadavky v integračních projektech se soustřeďují na specifikaci **komunikace mezi aplikacemi** (obsahově a výkonnostně), **nikoli** na koncového **uživatele**, a na co nejširší využití integračních standardů. Více v kapitole 8 Požadavky v integračních projektech.
- Ve složení projektového týmu jsou zastoupeny i **role specifické pro integrační projekty** - mezi ně patří např. zástupci integrovaných aplikací, vlastníci dotčených podnikových procesů, specialisté na integrační platformu a integraci aplikací. Více v kapitole 6 Projektový tým.
- Výběr vhodného produktu pro integrační platformu jako kritickou podnikovou aplikaci se řídí řadou kritérií, jak dokumentuje kapitola 14 Výběr integrační platformy a 17.5 Kritéria výběru integračního serveru. Integrační platforma se po implementaci stává podstatnou součástí všech projektů vývoje IS, které se potřebují integrovat na stávající podnikové aplikace. Proto je implementace integrační platformy často realizována jako samostatný projekt, který primárně podporuje integrační strategii.
- Analýza funkčních požadavků na integrovaný PIS je charakterizována rozpadem na řadu detailních **analýz integračních procesů a případů**, které představují základní analytickou abstrakci v integračních projektech. Více v kapitole 9 Integrační proces a integrační případ.
- **Výběr** vhodného **integračního postupu** je náročný vzhledem k desítkám aplikovatelných integračních přístupů a technologií. Široké spektrum integračních postupů je popsáno v kapitole 10 Integrační postupy a souvisejících přílohách.
- Jádrem řešení integračních projektů je integrační platforma, která se skládá z integračních komponent. Každá integrační komponenta podporuje určitou skupinu integračních postupů. Celkově tak integrační platforma poskytuje množinu integračních postupů, které si konkurují při řešení integračních případů. Postupné rozšiřování této množiny vede k postupné optimalizaci integračních cest v PIS, více v kapitole 11 Model komplexní integrované architektury PIS.
- Při návrhu řešení integračních procesů se uplatňuje snaha o **sdílení společné funkcionality**. Ta se projevuje na straně rozhraní formou sdílených integračních bodů pro více integračních případů i na straně integrační platformy v podobě společných funkčních bloků více procesů. Při modelování integračních procesů se využívají specifické designové vzory pro funkční celky i předávaná data. Více v kapitole 9.6 Sdílené prvky.

- Integrační platforma je primárně určena pro komunikaci mezi aplikacemi, tedy přes **systémová rozhraní**. V podstatné míře tak odpadá nutnost testovat uživatelského rozhraní. Naopak i malá chyba či výpadek při komunikaci mezi aplikacemi může vést k zásadní nekonzistenci mezi aplikacemi. Proto je **na testování** funkcionality, regresní a výkonnostní testy a testy obnovy systému kladen **velký důraz**. Testovací prostředí musí vedle integrační platformy zahrnovat i integrované aplikace nebo jejich emulátory. Pro testování je třeba mít připraveno velké množství reprezentativních testovacích dat. Více v kapitole 15 Testování.
- Provozování integrovaného systému vyžaduje stejně jako kterákoli kritická aplikace **neustálý dohled**. V integračních projektech je důležitá postimplementační podpora a řízení změn, neboť vedle požadavků na změny v integračních procesech a platformě se musí systém přizpůsobovat i změnám v integrovaných aplikacích. Více v kapitole 4.8 Provoz a údržba.

Tento soupis prokazuje první pracovní hypotézu, že i**ntegrační projekty mají specifické charakteristiky**.

Na integrační projekty lze aplikovat obecnou metodiku pro vývoj informačního systému, neboť integrovaný PIS je rovněž informačním systémem. To je prakticky potvrzeno i výsledky ankety (viz kapitola 3 Výběr mateřské metodiky). Sestavená metodika integračních projektů dokazuje druhou pracovní hypotézu, že **pro integrační projekty lze** (na základě popsaných odlišností) **nalézt specifické postupy**, které ji učiní efektivnější než obecnou metodiku. Důkaz efektivnosti je proveden logickou implikací, vycházející z času potřebného pro nalezení specifických postupů pro integraci aplikací při použití obecné metodiky (více v kapitola 1.2 Řešený problém).

Prezentovaná **integrační metodika** vychází z obecné metodiky MMDIS, z které přebírá rámec životního cyklu projektu a jeho dimenze. Vzájemná vazba metodik je popsána v kapitole 4 Životní cyklus integračního projektu. Mezi její **podstatné rysy** patří (v pořadí dle hlavních charakteristik integračních projektů):

- Vazba na integrační strategii jako hlavní dokument podniku k integraci aplikací.
- Specifikace integračního projektu v podobě business case dokumentu (viz kapitola 7 Business case) a typových požadavků na systém (viz kapitola 8 Požadavky v integračních projektech).
- Projektový tým mající jasně definované interní projektové skupiny a role s vymezenými odpovědnostmi (viz kapitola 6 Projektový tým).
- Proces výběru integrační platformy s multikriteriálním rozhodováním (viz kapitola 14 Výběr integrační platformy a 17.5 Kritéria výběru integračního serveru) s vazbou na referenční model integračního serveru (viz kapitola 12.1 Integrační server).
- Při analýze je integrační úloha rozdělena na integrační procesy (mající vazbu na podnikové procesy) a případy (viz kapitola 9 Integrační proces a integrační případ). Toto členění je dále základem pro návrh, vývoj a testování řešení integrační úlohy. Abstrakce integračního případu umožňuje výraznou atomizaci těchto fází podle identifikovaných integračních případů, což umožňuje realizovat tyto fáze řadou paralelních činností a snížit tak dobu jejich trvání.
- Výběr vhodného integračního postupu pro řešení integračního případu podle rozhodovacího stromu integračních postupů (viz kapitola 10.7 Rozhodovací strom integračních postupů), který je navázán na přehledové členění integračních postupů podle typu komunikace (přímá, zprostředkovaná) a typu rozhraní (datové, funkční, prezentační) v kapitole 10 Integrační postupy a souvisejících přílohách.
- Model komplexní integrované architektury PIS zobrazující integrační platformu jako soustavu integračních komponent, skrze kterou vedou integrační cesty, po kterých aplikace navzájem komunikují (viz kapitola 11 Model komplexní integrované architektury PIS). Model je otevřený pro další rozšiřování, jak je demonstrováno na

jeho rozšířené verzi (viz kapitola 13 Rozšířený model komplexní integrované architektury PIS).
- Referenční modely funkcionality integračních komponent integračního serveru a informačního podnikového portálu (viz kapitola 12 Modely integračních nástrojů).
- Proces testování je upravený pro potřeby integračních projektů (viz kapitola 15 Testování).

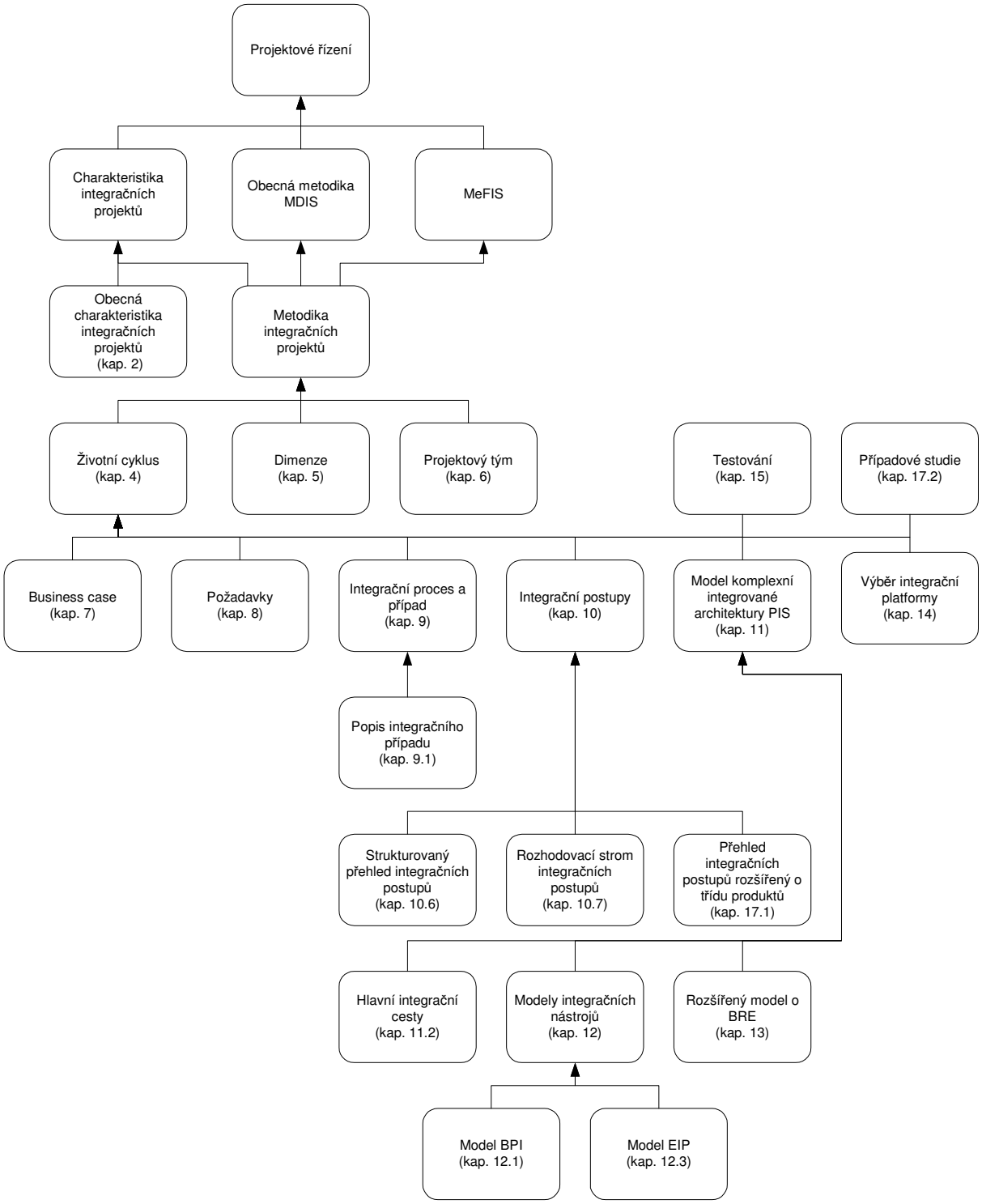
Podstatné části metodiky byly ověřeny na reálných komerčních projektech (viz kapitola 17.6 Případové studie). **Aplikace integrační metodiky** vedla k zefektivnění projektů zejména v těchto oblastech:

- Přesnému odhadování rozsahu projektu již v době uzavírání smlouvy (na straně dodavatele i zákazníka).
- Nastavení potřebné spolupráce a zafixování potřebných vstupů a očekávaných výstupů na počátku projektu.
- Přesnému plánování, monitorování a řízení projektu díky vazbě na integrační případy.
- Eliminaci planých diskuzí o způsobu řešení integračních případů.
- Významnému zkrácení dodacích lhůt projektů díky strukturovanému přístupu, který umožňuje efektivní (paralelní) rozdělování práce mezi jednotlivé týmy.

V kapitole 17.4 Integrační produkty je uveden přehled integračních produktů, který rozšiřuje přehled integračních technologií z kapitoly 10.6 Přehled integračních postupů a dokládá tak schopnost této části metodiky absorbovat aktuální vývoj v oblasti integračních technologií.

Hlavní přínosy této práce jako celku jsou uvedeny v úvodní kapitole 1.3 Cíle a přínosy. Obrázek 56 - Struktura přínosů práce (zdroj: autor) znázorňuje hierarchickou strukturu mých přínosů ke zkoumané problematice na těchto úrovních:

1. Přínos k projektovému řízení IS v doméně integrace aplikací v podobě hlubšího poznání integračních projektů a metodiky pro jejich realizaci.
2. Sestavení charakteristiky integračních projektů a provázání vytvořené metodiky s metodikou MMDIS a metodickým rámcem MeFIS do kompaktního celku metodik IS.
3. Vytvoření metodiky specifické pro vnitropodnikový integrační projekt definující jeho životní cyklus, dimenze a projektový tým.
4. Zkoumání fází integračního projektu a jejich definování z pohledu metodiky, zejména při:
   - sestavování business case dokumentu,
   - definici požadavků na systém,
   - analýze založené na integračních procesech a případech,
   - volbě integračních postupů,
   - návrhu architektury a výběru integrační platformy a
   - testování.
5. Rozpracování klíčových prvků metodiky detailněji:
   - u integrační procesu do detailní šablony jeho popisu,
   - u integračních postupů do rozsáhlého přehledu integračních koncepcí, přístupů, technologií a produktů s důrazem na vzájemné vazby, které jsou uplatněny v sestaveném rozhodovacím stromu pro výběr integračního postupu pro daný integrační případ,
   - u modelu komplexní integrované architektury PIS do koncepce hlavních integračních cest, modelů integračních komponent (BPI a EIP) a rozšířeného modelu s BRE.



obrázek 56 - Struktura přínosů práce (zdroj: autor)

Zásadní **osobní přínos** spatřuji v

- definici konceptuální rámce vnitropodnikového integračního projektu, tzn. vymezení jeho obsahu a struktury,
- systematickém zpracování metodiky integračního projektu,
- doplnění dimenze bezpečnosti (SEC) mezi dimenze projektu vývoje IS (implikuje zpětnou vazbu pro MMDIS) a její popsání pro integrační projekty,
- využití abstrakce integrační proces a integrační případ a jejich úzké provázání na fáze analýzy, návrhu a implementace,
- vymezení termínů integrační postup, koncepce, přístup, technologie, produkt, standard a definice jejich vztahů,
- systematické členění, pokrývající stávající integrační postupy, dle rozhraní vrstev aplikace a typu komunikace, které je dále prakticky využito v rozhodovacím stromu,
- začlenění integrace na úrovni prezentační vrstvy mezi tradičně chápané integrační postupy,
- sestavení rozhodovacího stromu pro výběr integračních postupu pro integrační případ,
- sestavení modelu komplexní integrované architektury PIS zachycujícího integrační cesty v PIS,
- identifikace potenciální role BRE v integrovaném PIS s pomocí rozšířeného modelu, komplexní integrované architektury PIS,
- sestavení univerzálních modelů BPI (a jeho využití při výběru BPI pro integrační platformu) a EIP,
- identifikace specifických rysů a požadavků testování v integračních projektech a
- ověření vytvořené metodiky na reálných komerčních projektech.

Tato práce otvírá možnosti pro **pokračování výzkumu** v několika dalších směrech. První oblastí jsou projekty specifické pro další typy komponent PIS, např. CRM, ERP, BI, jak je člení MeFIS. Stejným způsobem, jako je to provedeno v této práci, lze nalézat odlišné rysy těchto projektů a definovat pro ně specifické metodiky, které budou vycházet ze společné obecné metodiky MMDIS. Tím lze postupně vytvořit komplexní rodinu metodik pro implementaci celého PIS, která bude efektivnější než používání různorodých specifických metodik nebo samotné obecné metodiky pro oblasti, kde specifická metodika není k dispozici. Tato myšlenka celé rodiny metodik umožňuje využít jejich vzájemnou synergii postavenou na provázanosti metodik vycházejících ze společného základu. Porovnáváním a analýzou specifických metodik lze zpětně zlepšovat i společnou obecnou metodiku.

Druhou oblastí je rozvoj prezentovaného modelu komplexní integrované architektury PIS přidáváním dalších sdílených komponent, např. databáze metadat či řešení SSO, podobně jak je model rozšířen pro BRE, a rozpracování myšlenky integračních cest v PIS. V souvislosti s tímto modelem lze analyzovat tendenci dodavatelů integračních produktů o pokrytí celé škály integračních komponent vlastní kompaktní integrační platformou.

Třetí potenciální oblast zkoumání je dána velkou množinou integračních postupů, kde tato práce nastavila principiálně jednoduché a prakticky využitelné členění, které zahrnuje i integraci prezentační vrstvy. Tato integrace není v dostupných informačních zdrojích detailněji zpracována. Členění je navrženo univerzálně na elementární principech komunikace a vrstev aplikací, což umožňuje přidávat nové integrační postupy a zasazovat je do vztahů mezi stávajícími postupy. Nové integrační postupy mohou podle své podstaty vyžadovat rozšíření či úpravu navrženého rozhodovacího stromu integračních postupů či modelu komplexní integrované architektury PIS.

Závěrem konstatuji, že **práce splňuje oba cíle stanovené v úvodní kapitole**, tedy nalezení specifických charakteristik vnitropodnikových integračních projektů a sestavení specifické metodiky pro implementaci integrační platformy včetně navržených efektivních postupů a modelů. Její výsledky jsou prakticky využitelné v integračních projektech pro jejich efektivnější řízení.

17. Přílohy

17.1 Integrační koncepce

Při sestavování integrační strategie lze využít předpřipravené integrační koncepce (viz tabulka 21 - Přehled integračních koncepcí (zdroj: autor)) a kombinovat je se samostatnými integračními přístupy.

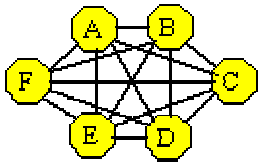
Integrační koncepce	Popis	Integrační úroveň	Užité integrační přístupy
Point-to-point (též na zakázku či špagetová architektura)	Aplikace jsou propojeny vazbami pro předávání dat a volání služeb bez centrálního řízení integrace. Pro každou vazbu je vytvořen specifický konektor.	datová služeb	Všechny přístupy přímé komunikace Replikace a transformace
Datová	Aplikace komunikují předáváním dat přes datová rozhraní.	datová	Přístupy využívající datové rozhraní
EAI	Komunikace mezi aplikacemi je řízena podle modelu podnikových procesů sestaveným na centrálním integračním serveru a probíhá v reálném čase. Aplikace jsou k serveru připojeny univerzálními konektory.	procesní	Integrační server
EDA	Spolupráce mezi aplikacemi je řízena událostmi a předáváním zpráv.	procesní	Zasílání zpráv
EII	Žádosti o data z jednotlivých aplikací jsou směrovány na centrální datové rozhraní, které je distribuuje do známých datových zdrojů. Takto získaná data konsoliduje a předává je aplikacím. Data jsou získávána v reálném čase.	datová	Virtuální databáze - XQuery gateway
SOA	Aplikace zpřístupňují svoji funkcionalitu prostřednictvím služeb. Při potřebě komunikovat používají společný registr služeb, kde naleznou partnera, který potřebnou službu poskytuje.	služeb	Service broker Webová služba
EIP	Prezentační vrstva více aplikací (zejména webových) je konsolidována do centrálního GUI. Portály jsou doplněny o vlastní obchodní logiku, která pracuje s daty získávanými prostřednictvím EAI, EII i přístupy datové koncepce.	prezentační	EIP Webový prohlížeč
DCE	Aplikace spolu komunikují vzdáleným voláním služeb decentralizovaně prostřednictvím společného middleware.	služeb	RPC
OGSA	Aplikace spolu komunikují prostřednictvím webových služeb,	služeb	Service broker Webová služba

	které obalují jednotlivé zdroje IS a vytváří tak servisně orientované distribuované prostředí.		Virtuální databáze - XQuery gateway Virtuální databáze - SQL gateway

tabulka 21 - Přehled integračních koncepcí (zdroj: autor)

17.1.1 Point-to-Point

Tato koncepce označuje decentralizovanou integraci mezi aplikacemi, kde spolu aplikace komunikují přímo. Rozhraní jsou vyvinuta na konkrétní případ komunikace s předem danou aplikací. Se zvyšujícím počtem aplikací je tato koncepce neudržitelná, neboť exponenciálně s počtem aplikací roste počet potřebných rozhraní (odtud synonymum špagetová architektura) - viz obrázek 57 - Topologie point-to-point koncepce (zdroj: [DIAMONDCLUSTER, 2001]), což vede ke zbytečným nákladům na vývoj i údržbu. Za předpokladu, že podnik má N aplikací a potřebuje je všechny navzájem propojit, znamená to celkem vytvořit N(N-1) rozhraní (za předpokladu že pro vzájemnou komunikaci bude stačit pouze jedno univerzální rozhraní), což pro průměrných 100 podnikových aplikací znamená vytvořit a udržovat 9900 rozhraní. Proto se aplikuje pouze v případech, kdy je ve finále integrováno velmi malé množství aplikací nebo kdy je integrována aplikace s velice specifickým rozhraním.



obrázek 57 - Topologie point-to-point koncepce (zdroj: [DIAMONDCLUSTER, 2001])

O této koncepci se zmiňuje např. [LINTHICUM, 2003], [LINTHICUM, 1999], [ACHARYA, 2003], [JUŘEK, 2004]. V [JANDOŠ, 2004] a [DIAMONDCLUSTER, 2001] je zmíněna pouze ve smyslu topologie, v [SCHMELZER, 2003] je označena jako custom (tedy integrace na zakázku) a naopak zdůrazňuje neuniverzálnost rozhraní.

Vybrané charakteristiky v informačních zdrojích:

- jednoduchá a rychlá implementace [SCHMELZER, 2003],
- neumožňuje růst [SCHMELZER, 2003],
- neudržovatelná [SCHMELZER, 2003],
- těsná vazba aplikací [SCHMELZER, 2003].

17.1.2 Datová koncepce

Datová koncepce pokrývá integraci mezi aplikacemi, které si předávají samotná data (bez příslušné business logiky) prostřednictvím datových rozhraní. V informačních zdrojích např. ([LINTHICUM, 2003], [LINTHICUM, 1999], [DIAMONDCLUSTER, 2001]) se tato koncepce často označuje jako integrace na datové úrovni, což ovšem není zcela přesné. Jak nepřímo upozorňuje např. [NOVOTNÝ, 2005], [ACHARYA, 2003] a [JANDOŠ, 2004], tak samotná data bez obchodní logiky lze předávat i s pomocí funkčních rozhraní, nikoli jen datových. Integraci na datové úrovni tak lze realizovat i jen přes funkční rozhraní, což však již nespadá do datové integrační koncepce.

[CAREY, 2005] identifikuje tři možné způsoby pro přístup k datům přes funkční rozhraní:

- webové služby - data jsou poskytována jednoduchou webovou službou (koncepce SOA),
- procedury - data jsou poskytována vzdálenou procedurou (koncepce EAI),
- deklarace - data jsou zpřístupněna deklarací dat v aplikacích, kde deklarace je využívána virtuální databází pro sestavování distribuovaných datových dotazů (koncepce EII).

Typická komunikace v datové koncepci probíhá dávkově, tedy s časovým zpožděním. V informačních zdrojích jsem tuto charakteristiku nenalezl explicitně zmíněnu, proto lze do této koncepce patrně zahrnout i koncepci EII, resp. EII můžeme chápat jako novou generaci datové integrační koncepce.

V některých informačních zdrojích (např. [PANCHA, 2002], [EBIZ01]) se můžeme setkat s užším pojetím datové koncepce, kde se míní pouze integrace pomocí technologie ETL.

Vybrané charakteristiky v informačních zdrojích:

- dávkově zaměřená [PANCHA, 2002],
- orientovaná na aktuální spojení (session) mezi aplikacemi [PANCHA, 2002],
- rozsáhlé využití metadat [PANCHA, 2002],
- efektivní přesun a transformace velkých objemů dat [PANCHA, 2002],
- integruje data do business pohledů [PANCHA, 2002],
- práce s relačními a multidimenzionálními databázemi [EBIZ01],
- iniciace se dle časového plánu [EBIZ01],
- vyžaduje minimální změny v aplikacích [DIAMONDCLUSTER, 2001], [SCHMELZER, 2003],
- nelze implementovat složitější obchodní logiku [SCHMELZER, 2003],
- obtížná integrace několikanásobných datových toků [SCHMELZER, 2003],
- pracuje primárně s historickými daty [LINDSTEDT, 2005].

17.1.3 EAI

EAI (Enterprise Application Integration) vnímáme v širší rovině jako způsob řešení integrace aplikací, v úzkém pojetí se jedná o koncepci založenou na procesním přístupu k analýze integračních problémů, integrační platformě typu integrační server a integraci prostřednictvím funkčních rozhraní aplikací. Oproti příkladu u point-to-point koncepce stačí v optimálním případě pro 100 integrovaných aplikací vytvořit a udržovat 100 rozhraní.

Podstatným rysem EAI je, že komunikace probíhá v reálném nebo skoro reálném čase (tato změna je označována v informačních zdrojích jako podnik s nulovou prodlevou (Zero Latency Enterprise). Vymezením EAI se zabývá např. [LINTHICUM, 2003], [NOVOTNÝ, 2005], [DIAMONDCLUSTER, 2001] a [KRISHNAN], jednotnou definici jsem však nenalezl. Protože SOA není s EAI v rozporu (naopak ji dále rozpracovává), je tato původní koncepce označována také jako **tradiční EAI** [LINTHICUM, 2003]. Posunem chápání obsahu EAI se zabývá [KRISHNAN], kde je SOA chápáno jako další generace EAI.

EAI přišla jako nová integrační generace ke stávající point-to-point a datové integraci, nedokázala je však dosud plně nahradit. Příčinu spatřuji v její neschopnosti přenášet a zpracovávat velké objemy dat v reálném čase.

Definici širšího pojetí EAI nalezneme ve [WHATIS] (převzato z [SKOGLUND, 2002]) - "EAI je termín pro plány, metody a nástroje určené pro modernizaci, konsolidaci a integraci informačních systémů v podniku. Typicky má podnik stávající legacy aplikace a databáze a chce je nadále použít zatímco přidává nebo migruje nové skupiny aplikací, které využívají Internet, e-commerce, extranet a další nové technologie".

[OVUM, 2001] definuje EAI jako "sadu technologií a procesů, které pomáhají automatizovat proces integrace aplikací vytvořených na zakázku a balíčkových aplikací tak, aby smysluplně spolupracovaly a vyměňovaly si informace ve formátech a kontextech, kterým každý rozumí".

[CZADEK, 2004] říká : “EAI je tvorba nových strategických řešení dané kombinací funkcionality existujících podnikových aplikací, komerčních balíků a nového kódu využívajících jednotný middleware“.

[STRÜVER, 2002] zmiňuje dvě jiné definice, které shrnuje následovně: "EAI sestává z metod a nástrojů. Metody sestávají z 'měkkých' prvků EAI jako systémová analýza, zkoumání podnikových procesů, restrukturalizace atd. Technické EAI nástroje zpřístupňují nově vyvinuté nebo optimalizované podnikové procesy."

Vybrané charakteristiky v informačních zdrojích:

- transakčně orientovaná [PANCHA, 2002], [PINKSTON, 2001],
- řízená událostmi [PANCHA, 2002],
- umožňující automatizaci podnikových procesů [PANCHA, 2002], [PINKSTON, 2001],
- zajišťuje transakční integritu a zasílání zpráv [PANCHA, 2002],
- řešení vyžadující intenzívní konzultace, komplexní implementace [PANCHA, 2002],
- podporuje asynchronní model [SCHMELZER, 2003],
- integrovaná business logika postavená na business pravidlech [SCHMELZER, 2003], [DIAMONDCLUSTER, 2001],
- prostředníkem řízená komunikace [SCHMELZER, 2003],
- problémy v synchronních modelech [SCHMELZER, 2003],
- proprietární řešení oproti SOA [SCHMELZER, 2003], [LINDSTEDT, 2005], [DIAMONDCLUSTER, 2001],
- pracuje primárně s aktuálními daty [LINDSTEDT, 2005],
- real-time přístup (poskytování informací v době, kdy jsou potřeba) [PINKSTON, 2001],
- plug-and-play přístup (možnost zaměnit aplikaci za jinou bez dopadu na provoz) [PINKSTON, 2001],
- zadávání dat pouze na jediném místě (v aplikaci) [PINKSTON, 2001],
- centralizovaná business pravidla [PINKSTON, 2001].

17.1.4 EDA

Koncepce EDA (Event Driven Architecture) je postavena na integrační přístupu zasílání zpráv. Nastane-li v aplikaci událost, která je relevantní dle business pravidel pro některou další integrovanou aplikaci, pak je informace o této události předána do této aplikace. Vzhledem k tomu, že je řízena událostmi v aplikacích, považujeme ji za procesně orientovanou. [HAUBRICH, 2005] tuto koncepci popisuje jako rovnováhu k SOA, kde EDA pracuje tlačným principem ("push" princip - v případě události "strčí" do další aplikace), zatímco SOA tažným principem ("pull" princip - v případě potřeby "tahá" služby z dalších aplikací). EDA považujme za koncepci, která byla absorbována koncepcí EAI.

17.1.5 EII

Koncepce EII (Enterprise Information Integration) "řeší integraci nesourodých datových typů z vícero datových zdrojů (vně i uvnitř podniku) v reálném čase, poskytující univerzální datovou vrstvu, která data zpřístupňuje pro aplikační využití" - definice převzata z [IMHOFF, 2004]. Data zůstávají fyzicky uložena pouze v datových zdrojích. Tato koncepce není nikterak nová, ale dostala možnost se reálně rozvinout až s příchodem dostatečně rychlých technologií, které umožňují zpracovávat větší množství dat v reálném čase. Koncepci EII se detailně věnuje [MORAGENTHAL, 2005a], [KERNOCHAN, 2003] a částečně i [FOURNIER-MOREL, 2004].

Vybrané charakteristiky v informačních zdrojích:

- iniciovaná dle aktuální potřeby nebo časového plánu [IMHOFF, 2004],
- poskytující data v reálném čase [IMHOFF, 2004],
- pracující s relačními databázemi i XML [IMHOFF, 2004],
- podporující inkrementální vývoj [IMHOFF, 2004].

17.1.6 EIP

Koncepce podnikových informačních portálů (Enterprise Information Portal) vychází z potřeby centralizovat podnikové informace v jednom jednotném uživatelském rozhraní a z rozvoje internetových technologií. Hlavním smyslem této koncepce je agregovat data z interních systémů a prezentovat je jednotnou a přehlednou formou prostřednictvím webových technologií. Data jsou získávána různými integračními způsoby - z přístupů koncepcí EAI, EII ,datové koncepce či přímo integrací prezentační vrstvy.

Stejným způsobem jsou naopak příchozí data z portálu předávána zpět do interních aplikací. Díky propojení EAI a EIP je přirozeně možné zpřístupnit nejen data aplikací, ale i jejich služby. Portál se tak stává bránou nejen k podnikovým informacím, ale i k podnikových službám. Podnikové portály mohou být využívány pro interní i externí prostředí podniku a díky svým nativním schopnostem mohou být neomezeně integrovány s dalšími portály.

17.1.7 DCE

DCE (Distributed Computing Environment) je koncepce založená na myšlence distribuované aplikace, která je sestavena z distribuovaných služeb. Tyto služby zahrnují (dle [SEI], [MALVEAU, 2000] a [LINTHICUM, 1999]):

1. Základní distribuované služby:
   a. RPC,
   b. adresářové služby - umožňují snadno identifikovat a přistupovat k distribuovaným zdrojům,
   c. časové služby - zajišťují mechanismus pro sledování času v distribuovaném prostředí,
   d. zabezpečovací služby - poskytují autorizaci, autentizaci a správu uživatelských kont,
   e. služby vlákna - podpora pro souběžné aplikace.
2. Služby sdílení dat:
   a. distribuovaný souborový systém,
   b. bezdisková podpora - umožňující užívat diskový prostor na serverech pro koncové stanice.

Od této koncepce se postupně ustupuje kvůli výkonnostním problémům ve prospěch EAI. [HOFFMANN, 2005] používá termín DCA (Distributed Component Architecture).

Vybrané charakteristiky v informačních zdrojích:

- objektově orientovaná [HOFFMANN, 2005],
- vyžadující homogenní technologii [HOFFMANN, 2005],
- s dlouhým vývojovým cyklem [HOFFMANN, 2005].

17.1.8 SOA

SOA (Service Oriented Architecture, někde ([SCHMELZER, 2003], [NEWCOMER, 2004]) též Integration - SOI) je koncepce (přesněji dle [SOA Blueprints] metodika návrhu) založená na principu univerzálních služeb, které integrované aplikace poskytují jiným aplikacím. [KRAFZIG, 2004] definuje SOA jako "softwarovou architekturu, která je postavena na klíčových konceptech aplikačního frontendu, službě, repository služeb a service busu. Služba se skládá z kontraktu, jednoho nebo více rozhraní a implementace". Implementace se skládá s business logiky a dat.

Služby v SOA mají přesně daná pravidla, jak mají vypadat a jak se mají chovat, a to umožňuje jejich univerzální integrovatelnost. SOA je integrace postavená na standardech. Rozdíl oproti tradičním službám aplikací je požadavek na schopnost splnit univerzální žádosti jiných aplikací - služba se musí chovat samostatně a být chráněná vůči nežádoucím stavům. Typickým a nejčastěji zmiňovaným příkladem standardizované služby je webová služba (ale SOA není jen o webových službách, jak poznamenává [STUMPF, 2004]).

[BUCHALCEOVÁ, 2005] definuje službu v SOA jako "abstraktní vrstvu mezi podnikovými procesy a technologií", [ERL, 2005] služby v SOA charakterizuje jako znovupoužitelné, sdílející definice služby, volně vázané, abstrahující vnitřní logiku, sestavitelné dohromady s dalšími službami, autonomní, bezestavové a vyhledatelné podle popisů.

Poskytované služby jsou evidované v registru dostupném všem integrovaným aplikacím. Aplikace požadující provedení služby se nejdříve dotáží do registru a v něm naleznou identifikaci aplikace i služby, kterou následně zavolají k provedení. Toto je princip service brokeru, který je uveden jako jeden z integračních přístupů.

SOA představuje model podniku, který poskytuje soubor služeb napříč celým podnikem. Tato architektura se skládá ze tří vrstev ([HOHPE, 2002a]):

1. Služby infrastruktury, které zahrnují:
    a. bezpečnost,
    b. administraci a monitorování,
    c. řízení vyjímek,
    d. logování,
    e. registraci a vyhledávání.
2. Neutrální podnikové služby, které zahrnují:
    a. service broker,
    b. notifikace,
    c. plánování,
    d. workflow.
3. Podnikové služby zaměřené na jádro podnikové aplikace, např. validace kreditních karet.

Integrace aplikací se již několik let výrazně zaměřuje a rozvíjí ve smyslu popsaných koncepcí EAI a SOA. Tyto koncepce si navzájem nekonkurují, spíše se do značné míry překrývají (viz také [BROWN, 2002]) - SOA je považována za novou generaci EAI. Současným preferovaným ideálem způsobu integrace aplikací je tedy integrační platforma (zprostředkovatel) podporující primárně integraci na úrovni funkčních rozhraní, které poskytují okolním aplikacím služby.

[PAPAZOGLOU, 2003b] popisuje koncepci rozšířené SOA (**ESOA**), která ze základních služeb tvoří kombinované služby (které navzájem spolupracují) a nad nimi řízené služby (s SLA (Service Level Agreement), certifikáty) - viz obrázek 58 - Rozšířená koncepce SOA (zdroj: [PAPAZOGLOU, 2003b]). [JANDOŠ, 2005] zmiňuje označení **ESA** (Enterprise Service Architecture), kde jsou podnikové služby kompozitními službami složenými z jiných služeb.

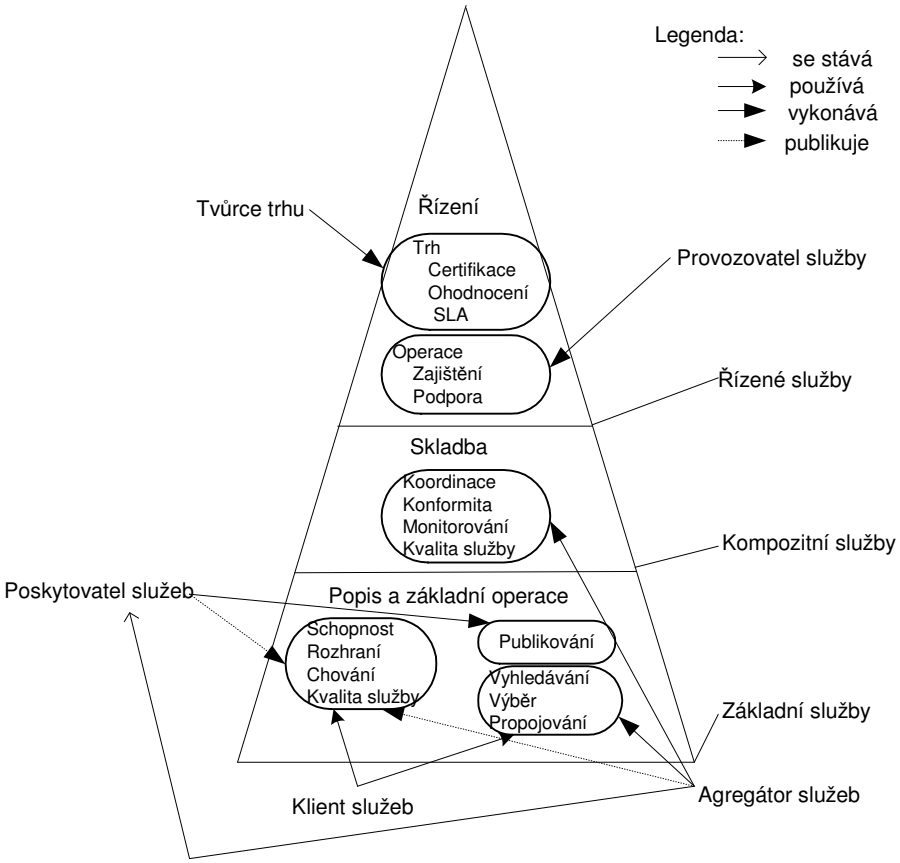
[PAPAZOGLOU, 2003b] rovněž popisuje různé způsoby outsourcování vývoje a provozování služeb:

- interní návrh a implementace služby,
- nákup/pronájem/platba za službu vyvinutou a provozovanou externě poskytovatelem služeb (ASP),
- outsourcování návrhu a vývoje služby externímu dodavateli, služba se realizuje interně,
- použití adaptérů/wrapperů pro legacy aplikace a databáze.

Koncepcí SOA se dále zabývá např. [SOA Blueprints], [PAPAZOGLOU, 2005], [ENDLICH, 2004], [BIEBERSTEIN, 2005], [NEWCOMER, 2004] a [BARRY, 2003]. [HALL, 2005] rozvíjí SOA do nového termínu **SOE** (Service Oriented Enterprise), který představuje podnikový informační systém založený na "chytrých" datech (data se sémantickým popisem v metadatech), "chytrou" infrastrukturou (zajišťující neselhávající komunikaci mezi aplikacemi postavou na zasílání zpráv) a "chytrých" službách (sémantické služby konfigurovaných jako webové služby).

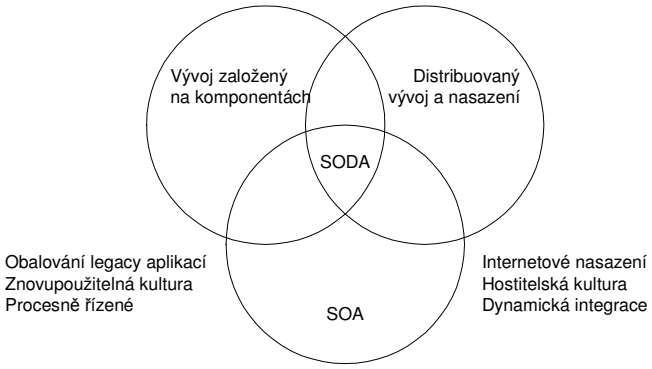
Vybrané charakteristiky SOA v informačních zdrojích:

- procesně orientovaná [HOFFMANN, 2005],
- s volnou vazbou mezi aplikacemi [HOFFMANN, 2005],
- pracující s heterogenními technologiemi [HOFFMANN, 2005],
- založeno na standardech [SCHMELZER, 2003],
- asynchronní i synchronní modely [SCHMELZER, 2003],
- nevyzrálost standardů [SCHMELZER, 2003],
- chybějící robustní bezpečnost a spolehlivost [SCHMELZER, 2003].



obrázek 58 - Rozšířená koncepce SOA (zdroj: [PAPAZOGLOU, 2003b])

Na koncepci SOA navazuje koncepce **SODA** (Service Oriented Development Architecture), která je určena pro vývojáře služeb dle SOA. SODA je založena na třech příbuzných koncepcích, jejichž provázanost ukazuje obrázek 59 - Vztah SOA a SODA (zdroj: [PLUMMER, 2003a]). Pro základní informace o SODA dále odkazuji na [PLUMMER, 2003a].



obrázek 59 - Vztah SOA a SODA (zdroj: [PLUMMER, 2003a])

17.1.9 OGSA

OGSA (Open Grid Services Architecture) přenáší myšlenky SOA do distribuovaného prostředí, je považována za nástupce DCE. [TREADWELL, 2005] definuje termín **grid** jako "systém, který se zabývá integrací, virtualizací a řízením služeb a zdrojů v distribuovaném, heterogenním prostředí, které podporuje množinu uživatelů a zdrojů (virtuálních organizací) napříč tradičními administrativními a organizačními doménami (reálnými organizacemi)". **Virtualizací** se míní transparentní přístup k distribuovaným výpočetním zdrojům. Službou gridu definuje stejný zdroj "webovou službu, která je navržena pro práci v prostředí gridu a splňuje požadavky gridu(ů), kterých se účastní". Vývojem a implementací gridových technologií se zabývá sdružení GGF (Global Grid Forum).

Podkladem pro OGSA je OGSI (Open Grid Services Infrastructure). Podle [BHATIA, 2005] vznikla z kombinací dvou infrastruktur - z vědecké komunity, která adoptovala framework Globus, což je sada nástrojů, která umožňuje vytváření a řízení virtuálních organizací), a z průmyslové, která stejně tak adoptovala webové služby. [BHATIA, 2005] vymezuje oblast zkoumání implementace P2P aplikací v OGSI ve třech směrech - distribuovaných výpočtech, spolupráci a sdílení obsahu.

[TUECKE, 2003] popisuje vztah mezi OGSA a OGSI takto - "OGSA spojuje klíčové grid technologie s mechanismy webových služeb k vytvoření distribuovaného systémového frameworku založeného na OGSI". OGSA představuje logickou střední vrstvu služeb na obrázku 60 - Konceptuální schéma OGSA (zdroj: [FOSTER, 2005a]) a člení služby do těchto skupin (dle [FOSTER, 2005a]):

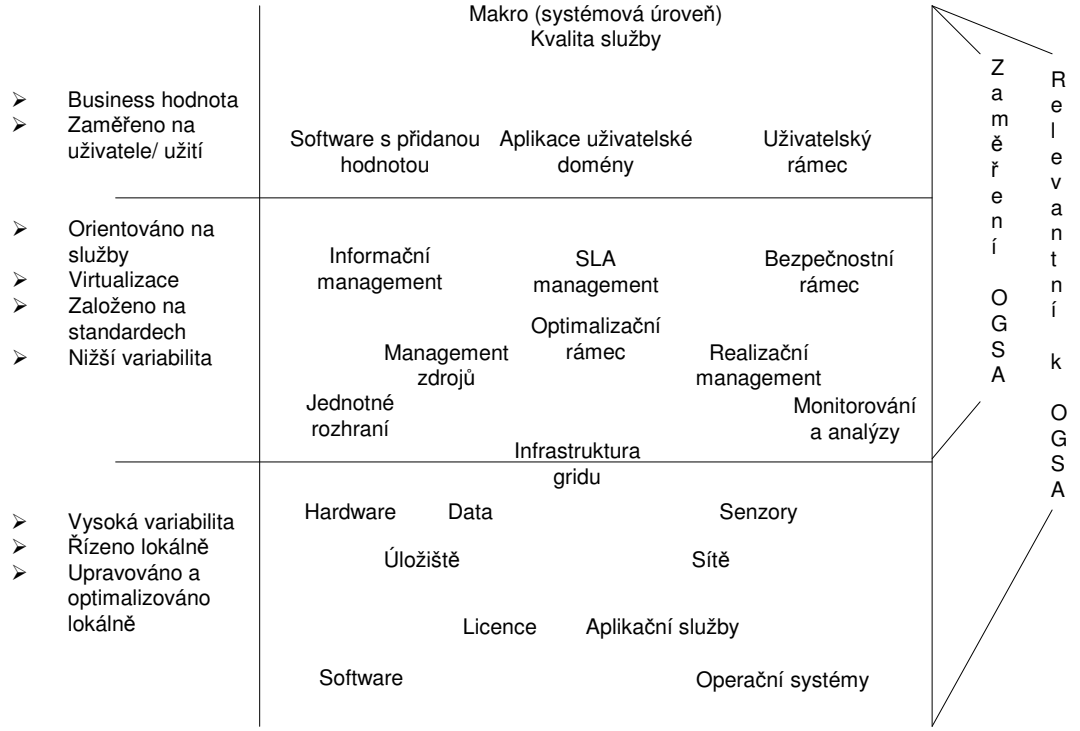
1. Řídící služby (Execution Management services) - zabývají se problémy vzniku instancí, řízením a ukončováním jednotek práce (což znamená buď OGSA aplikace nebo legacy aplikace).
2. Datové služby (Data services) - zabývají se přesunem, přístupem a aktualizací datových zdrojů.
3. Služby řízení zdrojů (Resource Management services) - pokrývá tři typy řízení:
    a. řízení zdrojů samotných (např. restartování severu),
    b. řízení zdrojů v rámci gridu (např. rezervování zdroje),
    c. řízení infrastruktury OGSA, která se sama skládá ze zdrojů (např. monitorování služby registrů).
4. Služby zabezpečení (Security services) - zajišťují aplikaci bezpečnostní politiky.
5. Služby pro vlastní řízení (Self-management services) - pro systémové komponenty, které se samy konfigurují, zotavují a optimalizují.
6. Informační služby (Information services) - umožňují efektivní přístup a manipulaci s informacemi o aplikacích, zdrojích a službách v prostředí gridu.

Datovými službami se více zabývá [ATKINSON, 2003], kde je popsána požadovaná transparentnost (virtualizace) datových služeb v oblasti heterogenity, umístění, jmenné identifikace, distribuce, replikace, vlastnictví a nákladů datových zdrojů. [RAMAN, 2003] navíc doplňuje ještě transparentnost paralelního zpracování operací nad datovými zdroji. Těmto požadavkům se dle [RAMAN, 2003] v současnosti nejvíce blíží integrační přístup virtuální databáze, který splňuje požadavky na transparentnost heterogenity a distribuce.

[RAMAN, 2003] definuje sadu služeb pro virtualizaci datových zdrojů:

1. Vyhledávací služba (Discovery service) - virtualizuje jmennou identifikaci a umístění.
2. Služba federativního přístupu (Federated Access service) - sestavuje jednotný virtuální pohled na distribuované datové zdroje.
3. Služba koordinovaného workflow (Workflow Coordination service) - deleguje operace na elementy gridu a přebírá zodpovědnost za jejich spolupráci včetně řešení jejich chybových stavů.

4. Služba řízení konzistence (Consistency Management service) - udržuje konzistenci mezi datovými zdroji.
5. Služba spolupráce (Collaboration service) - řeší sdílení dat mezi více uživateli včetně aktualizace.
6. Služba autorizace (Authorization service) - řeší SSO pro grid.
7. Služba řízení schémat (Schema Management service) - řeší problémy nejednotnosti datových schémat.
8. Replikační služba (Replication service) - odpovídá za udržování replikací a cache v rámci gridu.
9. Registr (Registry) - udržuje informace o zdrojích gridu.
10. Autentikační služba (Authentication service) - řeší podporu pro autentikační mechanismy.
11. Účetní a platební služba (Accounting/Billing service) - řeší zaúčtování a placení za operace nad datovými zdroji.
12. Notifikační služba (Notification service) - upozorňuje na změny v datových zdrojích.



obrázek 60 - Konceptuální schéma OGSA (zdroj: [FOSTER, 2005a])

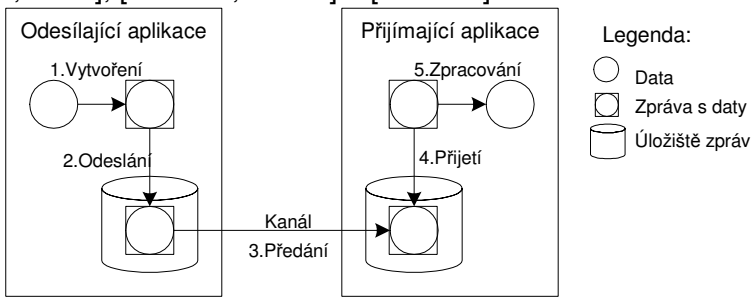
[POP2] nazývá infrastrukturu gridu, která umožňuje integraci a řízení služeb pro potřeby aplikací a systémů, **servisně gridovým busem** (SGB). Jeho architektura poskytuje zázemí pro registraci, vyhledávání, výběr a směrování, business pravidla, filtrování, agregaci, řešení výpadků, topologické mapování výskytů služeb, bezpečnostní politiku, transakce, vývoj a sestavování služeb. Svou funkcionalitou tedy odpovídá ESB (Enterprise Service Bus) - viz kapitola 17.3.2.2 Technologie funkčních rozhraní, ovšem v prostředí gridu.

17.2 Integrační přístupy

17.2.1 Přímá komunikace

Při přímé komunikaci rozlišujeme tyto komunikační modely (integrační přístupy):

1. U datových rozhraní:
   a. Replikace - zdrojová databáze předá vybraná data cílové databázi (platí i pro soubory) bez změny struktury.
   b. Databázový link - vybraná část vzdálené databáze se projevuje jako součást lokální databáze. Při dotazu funkční vrstvy na tato data v databázi je dotaz přesměrován na vzdálenou databázi a čeká se na odpověď.
2. U funkčních rozhraní:
   a. Jednorázová - komunikace proběhne zasláním jednoho požadavku (zprávy) a případně odpovědí na něj. Rozlišuje dva možné přístupy (také [HOHPE]):
      i. Volání vzdálené procedury (RPC - Remote Procedure Call) - volající aplikace požádá o provedení procedury ve volané aplikaci a počká na výsledek. Tento model je z principu synchronním příkladem komunikace.
      Jak již bylo zmíněno, nejčastěji se používá pro synchronní spolupráci, formou požadavek-odpověď (request-reply, také call-wait), kde klient musí čekat na odpověď serveru. Asynchronní volání požadavek-žádná odpověď (call-no wait) se využívá pouze vyjímečně. Synchronní volání vyžaduje, aby aplikace v roli serveru byla v době volání a zpracování dostupná, což je v distribuovaném prostředí nemožné zaručit. Proto musí mít technologické implementace tohoto přístupu dobře zpracovány ošetření chybových stavů, přesměrování na jiný server, časování požadavků atd.
      O RPC detailně píše např. [LINTHICUM, 2003], [SEI]. Název RPC pro tento integrační přístup není jednotný (používá se z historických důvodů), např. [JANDOŠ, 2005] používá termín volání funkce. Tento přístup v současné době zastřešuje několik integračních technologií, jak si popíšeme dále.
      ii. Zasílání zpráv (Messaging) - volající aplikace vyšle zprávu (message), která je volanou aplikací zachycena, vyhodnocena a cílová aplikace provede požadovanou službu (tedy v zásadě volaná aplikace rozhoduje, jak bude reagovat na danou zprávu). Volající aplikace nečeká na odezvu. Principiálně se jedná o asynchronní koncepci, neboť reakce aplikace není z pohledu volané aplikace definovaná. Synchronní spolupráci lze "simulovat" tím, že reakce volané aplikace je volající aplikaci předem známa a tudíž ji může očekávat. Jelikož se však není tato technologie primárně zaměřena na synchronní zpracování, je problematické její využití při transakčním zpracování.
      O zasílání zpráv (včetně zprostředkovaného) se detailně dočteme v [JUŘEK, 2004], [HOHPE, 2003b] a [HOHPE].



obrázek 61 - Schéma zasílání zpráv (zdroj: [HOHPE])

   b. Konverzační - komunikace probíhá formou dialogu, tedy postupným odpovídáním na několik požadavků, které se postupně formulují na základě předchozích odpovědí. Tento model je typický pro transakční zpracování. V případě transakčního zpracování se používá rozšířené TRPC (Transactional RPC).
3. U prezentačních grafických rozhraní (obalová prezentační rozhraní jsou vždy zprostředkovaná):
   a. Žádost o stránku (page request) - volající aplikace požádá volanou aplikaci o sestavení obrazovky dle parametrů a poté ji převezme. Tento postup nazývá [ATTACHMATE], bez vztahu k webovým technologiím, tzv. refacingem. [ROSENBLOOM, 2004], který se zabývá možnostmi sestavování stránek legacy aplikací, nazývá stránku portletem. Tento termín je však specificky užívaný v portálových řešeních.

17.2.2 Nepřímá komunikace

Komunikační modely s využitím zprostředkovatelů rovněž rozdělíme podle kategorií rozhraní, neboť i zde platí, že spolu mohou komunikovat jen rozhraní stejné kategorie:

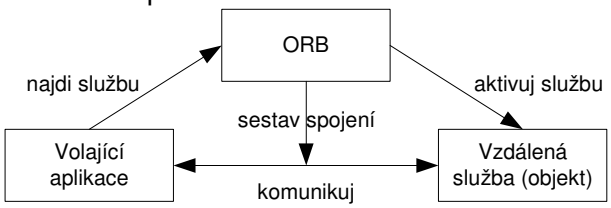
1. U datových rozhraní se rozvíjí komunikační model:
   a. Replikace do:
      i. Replikace a transformace - data ze zdrojové databáze jsou transformovány do struktur cílové databáze pomocí externího nástroje (nejsou tedy transformovány ani ve zdrojové, ani v cílové aplikaci).
      O možnostech replikace a transformací píše [SEACORD, 2003].
      ii. Datový sklad (data store) - data ze zdrojových databází jsou transformována do univerzálních struktur (typickým příkladem je struktura zákazníka a jeho služeb), dočasně uložena v datovém skladu (nezávislá databáze) a poskytována cílovým aplikacím dle potřeby z datového skladu v univerzální podobě. Výhodou je, že zdrojové i cílové aplikace musí umět pracovat pouze s univerzální strukturou dat, kterou podporuje datový sklad. Nevýhodou je nutnost udržovat data v datovém skladu neustále aktuální.
   b. Databázového linku do
      i. Virtuální databáze (též federated database, database gateway, datový hub, distributed query engine) - data ze zdrojových databází jsou dostupné ve formě jedné společné databáze. Oproti datovému skladu jsou zdrojová data dostupná v původní struktuře datových zdrojů. Zároveň nejsou replikována do centrální databáze, ale zůstávají pouze ve zdrojových datových zdrojích. Virtuální databáze agregují připojené databáze na logické úrovni a poskytují navenek jedno univerzální rozhraní.

   Zprostředkovatel datové integrace by měl zajišťovat procesy agregace, konsolidace, transformace, filtrování, validace i čištění dat.
2. U funkčních rozhraní se rozvíjí komunikační model:
   a. Zasílání zpráv do:
      i. Message Oriented Middleware (MOM) (v [HOHPE] označen jako Message Bus) - zajišťuje předávání zpráv mezi aplikacemi. Odeslaná zpráva je zařazena do fronty zpráv čekajících na zpracování u cílové aplikace. Umožňuje point-to-point i publish-subscribe komunikaci.
      [HOHPE] definuje MOM jako "systém umožňující komunikaci mezi oddělenými systémy prostřednictvím samostatných kousků informací (zpráv)přenášených skrze individuálně adresované kanály. Kanály mohou vytvářet fronty zpráv (asynchronní komunikace) nebo taky ne (synchronní komunikace)."

ii. Message broker (MB) - rozšiřuje možnosti MOM před předáním do cílové aplikace o
   - transformaci zpráv - mění se formát zprávy, zprávy mohou být rozdělovány či sloučeny dle potřeby
   - zpracování zpráv dle business pravidel - implementují se jednoduchá pravidla, která pracují s obsahem zprávy
   - inteligentní směrování - spočívá v nalezení vhodného příjemce zprávy dle definovaných pravidel

b. Volání vzdálené procedury do:

i. Object Request Broker (ORB) - tento model pracuje s funkčními rozhraní objektovými. ORB vyhledává cílovou službu (objekt) a sestavuje spojení mezi volající a cílovou aplikací.

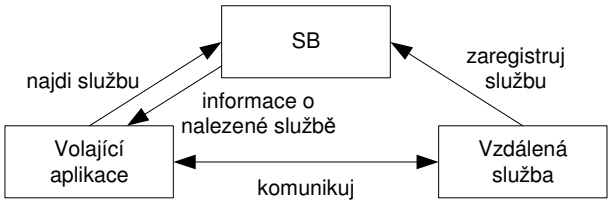


obrázek 62 - Object request broker (zdroj: [SEI])

S ORB souvisí širší spektrum služeb (vybráno ze specifikace CORBA [SEI] a [LINTHICUM, 1999]):

- Objektové služby - základní služby pro vytváření distribuovaných aplikací:
  - Naming Service - umožňuje přiřadit objektu název.
  - Event Service - podporuje asynchronní zasílání zpráv.
  - Lifecycle Service - definuje konvence pro životní cyklus objektů.
  - Persistence Service - poskytuje prostředky pro správu trvalých stavů objektů.
  - Transaction Service - podporuje transakční zpracování,
  - Concurrency Service - podporuje současný přístup k objektu z více klientů.
  - Relationship Service - podporuje správu vztahů mezi objekty.
  - Externalization Service - definuje protokoly a konvence pro realizaci objektů.
- Společné příslušenství - specifické služby využitelné pouze v některých případech:
  - Uživatelské rozhraní.
  - Informační management.
  - Systémový management.
  - Management úloh.
- Aplikační objekty - služby specifické pro určitou aplikaci nebo třídu aplikací.

ii. Service Broker (SB) - principiálně shodný s ORB, SB spravuje místo objektů služby.



obrázek 63 - Service broker (zdroj: [BARRY, 2003])

Obrázek 63 - Service broker (zdroj: [BARRY, 2003]) znázorňuje užití SB v koncepci SOA. Volající aplikace je v roli uživatele služby, vzdálenou službu nabízí poskytovatel služby a SB je v roli registru služeb.

Na příkladu webových služeb komunikace probíhá následovně:

1. Poskytovatel služby (Service Provider) zašle WSDL specifikaci služby Service brokeru, který ji zveřejní.
2. Žadatel o službu (Service Requester) požádá Service broker o službu na základě názvu služby nebo parametrů.
3. Service broker vybere poskytovatele služby a zašle informace žadateli.
4. Žadatel na základě informací od Service brokeru sestaví SOAP požadavek (formát zprávy a identifikace poskytovatele) a zašle jej poskytovateli.
5. Poskytovatel sestaví odpověď na požadavek a zašle SOAP/XML odpověď.

c. Konverzační do:

i. Transakční monitor (TP - Transaction Processing Monitor) - je určen pro zpracování několika operací (zpravidla v různých aplikacích) v jedné transakci. Umožňuje definovat business pravidla mezi operacemi.

[LINTHICUM, 2003] shrnuje hlavní služby TP:

- garantují integritu transakcí,
- řídí zdroje a běh aplikačních služeb.

ii. Aplikační server (AS) - jsou rozšířením možností TP. Vznikly primárně pro podporu webových aplikací, které potřebují vykonávat služby ve vzdálených aplikacích. AS funguje v roli jediného poskytovatele všech služeb. Tyto služby se skládají z volání vzdálených aplikací (s podporou transakčního zpracování), která jsou doplněna dalšími business pravidly modelovanými přímo na aplikačním serveru ve standardním programovacím jazyce. AS podporují komunikační modely zasílání zpráv i volání vzdálené procedury.

[JANDOŠ, 2004] definuje AS jako "hostitelský proces realizovaný SW produktem, představujícím prostředí pro zpracování aplikací (napsaných v určitém jazyce / programátorském prostředí - často J2EE). Pro zpracování zajišťuje různé služby jako např. bezpečnost, možnost využívat v rámci aplikace různé autonomní zdroje datové (např. databáze) nebo aplikační (tj. samostatné autonomní aplikace), řízení transakčního zpracování (s vlastnostmi ACID - nedělitelnost, konzistence, izolovanost, stálost) nad těmito autonomními zdroji".

AS se zabývá rovněž [PAPAZOGLOU, 2005] a [BASS, 2003].

iii. Integrační server (BPI - Business Process Integration server) - doplňuje služby aplikačního serveru o modelování a monitorování podnikových procesů.

[JUŘEK, 2004] používá termín integrační broker, [LINTHICUM, 2003] termín integrační server, [JANDOŠ, 2004] procesní integrační server (BPM/BPI (Business Process Management/ Integration) server), [JOHANNESSON, 2000] procesní broker.

3. U prezentačních rozhraní grafických (obalované jsou shodné s funkčními) se rozvíjí komunikační model:

a. Žádost o stránku do:

i. Webový server - je společným poskytovatelem webové prezentační vrstvy pro více aplikací. Žádosti o sestavení stránky buď plní sám dle

definovaných business pravidel a repository obsahu, nebo si stránky vyžádá od vzdálených aplikací, a následně je předá uživateli.

ii. Podnikového informačního portálu (EIP - Enterprise Information Portal) - rozšiřuje možnosti webového serveru a poskytuje framework pro tvorbu a provozování prezentační vrstvy (omezenou na webové technologie). EIP pracuje s daty a službami vzdálených aplikací, což mu umožňuje integrovaný AS. Dále využívá vlastní nebo vzdálené repository pro sestavování obsahu stránek (CMS - Content Managment Server) a vzdáleným voláním si od aplikací může vyžádat hotové stránky, které integruje do výsledné stránky prezentované uživateli. Vedle sestavování výsledné stránky se EIP stará o udržování sdílených informací (např. data o uživateli), které jsou relevantní při sestavování stránky a propaguje je do vzdálených aplikací.
Portálově orientovanou integrací se detailně zabývá např. [LINTHICUM, 2003].

iii. Webového prohlížeče (WB - web browser) - narozdíl od předchozích dvou případů se jedná o integraci na straně klienta, nikoli serveru. Webový prohlížeč si dle instrukcí v hlavní stránce vyžádá zaslání částí stránky z více vzdálených serverů (chápejme je jako aplikace) a provede jejich zakomponování do výsledné stránky.
Výše uvedené modely jsou popsány přímo na webových technologiích a to z toho důvodu, že jsou takto široce rozšířeny a implementace na jiných technologiích jsou zastaralé. Popis jednotlivých integračních přístupů není webovou technologií zkreslen.

Obdobné členění navrhuje i [PUSCHMANN, 2004], který tuto část integrace rozděluje na integraci na straně webového prohlížeče a na straně serveru.

4. U prezentačních rozhraní obalových:

i. Snímání obrazovek (Screen Scraping) - u těchto rozhraní je komunikace vždy zprostředkovaná, neboť obal původní aplikace tvoří třetí aplikace. V důsledku pak komunikace může být i dvojnásobně zprostředkovaná - nejprve aplikací vytvářející obal původní aplikace a následně některým ze zprostředkovatelů uvedených u funkčních rozhraní. Obal simuluje činnost uživatele definovanou sekvencí kroků prováděných v dané aplikaci. Zároveň umožňuje z výsledné obrazovky číst vybraná data.
[SKOGLUND, 2002] definuje snímač obrazovek jako "nástroj, který čte prezentovaná data z hostitelského informačního systému, reformátuje je dle potřeby a prezentuje je v novém uživatelském rozhraní. Původní prezentační vrstva zůstává nezměněna. Nástroj musí být rovněž schopen předávat vstupy z nového uživatelského rozhraní tak, aby byly zpracovány hostitelským informačním systémem." K této definici pouze doplním, že se nemusí jednat pouze o uživatelské rozhraní, ale v našem případě spíše o systémové rozhraní k dalším aplikacím.
Tento postup nazývá [ATTACHMATE], bez vztahu k webovým technologiím, tzv. repurposingem.

Tento způsob integrace se zpravidla volí v případě, kdy nemá integrátor jiné alternativní řešení, zpravidla u black-box aplikací, neboť se jedná vždy o přímou a nesystémovou integraci mezi dvěma aplikacemi, která je poměrně náročná na vývoj i údržbu.

17.3 Integrační technologie

17.3.1 Přímá komunikace

Přímou komunikaci můžeme bez ohledu na zvolený integrační přístup charakterizovat z pohledu technologií:

- použitým jazykem - např. XML,
- přenosovým protokolem - např. HTTP, FTP, SOAP.

Vedle těchto standardů se používá řada proprietárních protokolů a jazyků dle konkrétního produktu.

17.3.1.1 Technologie datových rozhraní

1. Replikace - replikační technologie rozdělíme v souladu s možnostmi datových rozhraní na:
    a. Souborové - přenos souborů mezi aplikacemi lze realizovat prostřednictvím:
        - možností operačního systému (OS) - pokud je OS zdrojové aplikace kompatibilní s OS cílové aplikace,
        - specializované nadstavby operačního systému podporující standardizovaný přenos souborů - např. přenos za pomoci FTP nebo HTTP serveru a klienta, nebo jejich zabezpečené varianty,
        - specifické (proprietární) nadstavby pro přenos souborů.
    b. Databázové - k databázi lze přistupovat přes (viz také [LINTHICUM, 1999]):
        - nativní databázový middleware,
        - standardizované rozhraní (CLI - Call Level Interface), kde se setkáváme s ODBC (Open Database Connectivity) či JDBC (Java Database Connectivity).

    Databázová replikace probíhá vyexportováním dat ze zdrojové databáze a následným importem dat do cílové databáze. Přitom může být prováděna transformace dat na straně zdrojové i cílové databáze.

    Existuje několik způsobů iniciace replikace:
        1. Cílová aplikace zjistí, že potřebuje data a požádá o ně, zpravidla zavoláním databázové procedury na straně zdrojové aplikace. Tato procedura zajistí předání dat cílové aplikaci.

           Ačkoli se zde volá procedura, zahrnuji tento způsob místo do funkčních rozhraní do datových. Je to z toho důvodu, že tento typ procedury neobsahuje prakticky žádnou obchodní logiku, pouze fyzicky zajišťuje přenos dat a oproti RPC zde funguje pouze jako spouštěč replikace. V opačném případě bychom nemohli o čistě datové integraci uvažovat, neboť data se sama od sebe přenášet nemohou.
        2. Zdrojová aplikace zjistí (např. pomocí databázového triggeru), že se obsah databáze změnil a předá změny cílové aplikaci.
        3. Nastane externí událost (např. podle daného časového plánu), která iniciuje přenos dat.
2. Databázový link - nerozvinuto do samostatných technologií, známo jako technologie databázového linku.

17.3.1.2 Technologie funkčních rozhraní

V oblasti technologií přímé komunikace prostřednictvím funkčních rozhraní soupeří hlavní tři vývojové platformy - J2EE (Java 2 Enteprise Edition), .NET a CORBA. Nemá význam zde podrobně rozebírat možnosti těchto platforem a navzájem je porovnávat, neboť objektivní a smysluplné srovnání je velmi obtížné. [NATIS, 2003a] poskytuje orientační porovnání J2EE a .NET - viz tabulka 22 - Základní srovnání .NET a J2EE (zdroj: [NATIS, 2003a]). O .NET pojednává např. [RUBIOLO, 2005].

.NET	J2EE
produkt jednoho dodavatele, chybí blueprint	konkurenční dodavatelé a implementace certifikace, komunity, referenční implementace
více vývojových jazyků	jeden vývojový jazyk
pro jeden operační systém	různé operační systémy
předintegrované prostředí	vyžaduje systémovou integraci
nepovinné objektově orientované znalosti	vyžaduje objektově orientované znalosti
preferuje tlustého klienta	preferuje tenkého klienta
nové, problematický přechod	rozsáhlá instalovaná základna
webové služby jako vlastnost	webové služby jako vrstva

tabulka 22 - Základní srovnání .NET a J2EE (zdroj: [NATIS, 2003a])

Uvedeme si zde pouze jejich technologie, které nás zajímají z pohledu integrace (vedle nich se využívají i rozhraní pro přístup k objektům / komponentám, jako např. OLE DB, ADO. NET, JDO (Java Data Objects)).

[STRÜVER, 2002] se zabývá vztahem objektů a komponent - komponenty rozšiřují objektový přístup. **Komponenty** jsou chápány jako striktně oddělené, samostatné objekty pro distribuované prostředí. Objekty a komponenty nejsou v informačních zdrojích explicitně rozlišovány, případně jen velmi vágně. Proto budu nadále používat termín objekt, neboť komponenta má i přenesený význam ve smyslu součásti něčeho.

1. Volání vzdálené procedury:
    a. J2EE - Java RMI (Remote Method Invocation), EJB (Enterprise Java Beans)
    b. .NET - .NET Remoting
    c. CORBA - CORBA RPC

   Vedle těchto technologií se užívá i volání vzdálených databázových PL/SQL procedur, které ovšem nejsou objektově orientované.
2. Zasílání zpráv
    a. J2EE - JMS (Java Messaging Service)
    b. .NET - MSMQ (Microsoft Message Queue)
    c. CORBA - CORBA Messaging Services

   Technologie zasílání zpráv má některé nevýhody (viz také [HOHPE]):
    - Komplexní programový model - zpracování je založeno na událostech, což znamená, že vývojář musí systém naprogramovat nikoli pomocí vzájemného volání jednotlivých funkcí a procedur, ale jako reakce na určité události. Tento přístup je výrazně náročnější vzhledem k ošetření možných situací v systému.
    - Message brokery primárně nepodporují následující funkce, které je nutno řešit vlastním rozšířením:
        - zajištění sekvenčního pořadí zpráv (případně řešení priority zpráv),
        - synchronní zpracování zpráv,
        - zajištěním doručenosti zprávy,
        - podpora transakčního zpracování.
    - Výkonnost - zpracování zpráv zprostředkovatelem pochopitelně přináší nárůst přenášených dat, neboť obsahují i informace určené pro broker. Navíc tyto systémy nepodporují velké objemy zpráv.
    - Proprietární protokoly - pro přenos zpráv integrační platformy používají své vlastní proprietání protokoly, což omezuje flexibilitu systému.
3. Konverzační - pro konverzační integrační přístup si konkurují technologie:
    a. CORBA Transaction Services
    b. JTS (Java Transaction Services)
    c. .NET Transaction Services

Vedle těchto technologií byla vyvinuta standardizovaná technologie - **webová služba** (web service), která podporuje současně asynchronní i synchronní komunikaci. Lze ji tedy užít pro volání vzdálené služby i pro zaslání zprávy (resp. dokumentu, jak je častěji zmiňováno v souvislosti s webovými službami).

Webové služby nabízejí vícero možností využití dle [KAYE, 2003]:

- interní systémové webové služby - uvnitř jednoho systému, resp. aplikace,
- interní webové služby - uvnitř podnikového informačního systému,
- externí webové služby - vně podnikového informačního systému.

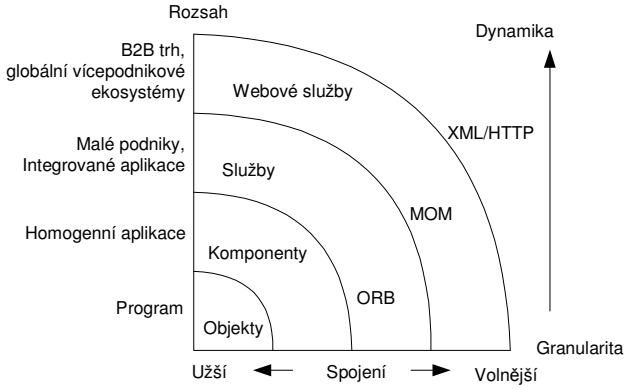
Z těchto skupin nás zajímají pro vnitropodnikovou integraci pouze interní webové služby.

Webová služba je příkladem služby v SOA. Pro vývoj webových služeb se užívá WSPP (Web Service Producer Platforms), architektura založená na webových službách se nazývá WSA (Web Service Architecture). Pro definici webových služeb je definován WSDL (Web Service Definition Language).

Vedle toho existují i další standardy - WSRP (Web Services for Remote Portals), WSUI (Web Services User Interface), WS-Security, SAML (Security Assertions Markup Language) (viz např. [CLAUNCH, 2003]). XLANG/XAML ((Transaction Authority Markup Language) specifikuje transakční chování a podporu 2PC (two-phase commit) protokolu (viz [FEUERLICHT, 2001]).

Principy webových služeb jsou vysvětleny v řadě informačních zdrojů, např. [KAYE, 2003], [NEWCOMER, 2002], [TIDWELL, 2001], [CERAMI, 2002], [BARRY, 2003], [BURANSKÝ, 2005]. [COHEN, 2004] se zabývá optimalizací a testováním webových služeb. V [PYROVOLAKIS, 2003] je popsáno, jak se webové služby využívají při modelování, resp. podpoře podnikových procesů.

Vztah webové služby k předchozím technologiím znázorňuje obrázek 64 - Vztah webové služby ke svým předchůdcům (zdroj: [WAGNER, 2003]).



obrázek 64 - Vztah webové služby ke svým předchůdcům (zdroj: [WAGNER, 2003])

V SOA se vyžaduje zasílání zpráv s plnou garancí doručení. Pro webové služby se připravují dvě specifikace (uvádí [STUMPF, 2004]):

- Web Services Reliability – reprezentovaná společnostmi Fujitsu, NEC, Oracle, Sonic Software, Sun Microsystems a dalšími,
- WS-Reliable Messaging – reprezentovaná společnostmi IBM, Microsoft.

Druhou významnou charakteristikou zasílání zpráv v SOA je směrování zpráv podle jejich obsahu, tedy např. pokud bude zpráva obsahovat v políčku segment hodnotu 1, pak zpráva bude odeslána jinam než v případě, že hodnota bude 2.

Několik informačních zdrojů (např. [SVÁTEK, 2004], [ARPINAR, 2005]) se zabývá rozšiřováním možností vyhledávání, zejména externích, webových služeb pomocí jejich zdokonaleného a automatizovaného sémantického popisu a vyhledávání. Řešení této úlohy se může postupně promítnout i do interních webových služeb.

V [CIMETIERE, 2003] je k dispozici průzkum z února 2003 o pronikání a využívání webových služeb, v [LAWLER, 2005] jsou k dispozici výsledky studie několika finančních firem při zavádění webových služeb.

17.3.1.3 Technologie prezentačních rozhraní

1. Žádost o stránku - volaná aplikace musí být schopna vytvořit požadovanou obrazovku, resp. její popis. Stránka může být statická nebo dynamická (nemá pevně daný obsah). Pro tvorbu dynamických webových stránek se používá několik technologií, mezi nimi:
    a. ASP (Active Server Pages),
    b. JSP (Java Server Pages),
    c. PHP (PHP: Hypertext Preprocessor).

    Pro vygenerování stránky stačí volající aplikaci zaslat identifikaci stránky (např. URL) se vstupními parametry volané aplikaci a ta požadovanou stránku sestaví. Dokáže-li volající aplikace přečíst formátování takto získané stránky, pak ji zakomponuje do vlastní výsledné stránky pro uživatele.
    Elementárním příkladem volající aplikace může být webový browser či klasická podniková aplikace. Pokud volající aplikace přijatou stránku již dále nekombinuje ani neupravuje, pak se jedná o přímou komunikaci, v opačném případě o komunikaci zprostředkovanou.

17.3.2 Zprostředkovaná komunikace

17.3.2.1 Technologie datových rozhraní

1. Replikace a transformace - pro transformaci (změnu formátu i struktury dat, aplikaci jednoduchých business pravidel) dat mezi zdrojovým a cílovým datovým úložištěm se používají tyto technologie:
    a. Extract-Transform-Load (ETL) - technologie umožňující transformace definovat pomocí vizuálního prostředí a monitorovat běh transformování. Vedle toho bývá doplněna dalšími technologiemi, např. čištěním dat. Umí zpracovávat soubory i data z databáze.
    b. Transformace na zakázku - alternativně vedle ETL lze transformace dat naprogramovat přímo do databázových procedur (obdobně i pro data v souborech). Jsou-li tyto procedury odděleny od zdrojové a cílové databáze (např. v DSA) , potom se jedná o zprostředkovanou komunikaci.
    [SEACORD, 2003] označuje tuto technologii jako integraci pomocí skriptů.

    Jako doplněk k těmto technologiím se v některých případech aplikuje i **dočasné datové úložiště** - Data Stage Area (DSA), do kterého se zkopírují vybraná data ze zdrojové databáze a veškeré transformační operace probíhají v DSA. DSA se používá pro snížení zátěže zdrojového systému. [NOVOTNÝ, 2005] charakterizuje data v DSA jako detailní, nekonzistentní, neobsahující historii, měnící se s každým snímkem ze zdrojové aplikace a jsoucí ve shodné struktuře se zdrojovým systémem.
2. Datový sklad - tento integrační přístup se promítnul do dvou technologií:
    a. Operační datový sklad - ODS (Operational Data Store) se skládá z vlastní databáze a rozhraní, které poskytuje transformační procedury. [NOVOTNÝ, 2005] charakterizuje data v ODS jako konsolidovaná, konzistentní, neobsahující historii, měnící se s každým snímkem ze zdrojové aplikace a jsoucí subjektově orientovaná (data jsou rozdělována podle obecných typů, nikoli dle aplikací).
    b. Datawarehouse (DWH) - obdoba ODS, která navíc obsahuje historická data. DWH se používá nejen pro integrační účely, ale zejména pro reportování. Často nemusí být DWH k integračním úlohám ani využit a plní úlohu datové základny pro reportovací nástroje. [NOVOTNÝ, 2005] charakterizuje data v DWH jako subjektově orientované, integrované, stálé a časově rozlišené.

Je-li potřeba s daty z DWH pracovat ve vzdálených lokalitách, pak vznikají tzv. datová tržiště (DMA - Data Mart), což jsou části DWH distribuované právě v těchto lokalitách pro lokální užití. Vzájemná konzistence mezi datovými tržišti se řeší dvěma základními architekturami DWH:

- Nezávislá datová tržiště - části DWH spolu komunikují v topologii sběrnice (bus).
- Závislá datová tržiště - primárním zdrojem dat je centrální část DWH (hub), z které jsou data poskytována do datových tržišť.

Z hlediska implementace jsou tyto architektury rovněž rozdílné, neboť vybudování centralizovaného DWH je z počátku podstatně náročnější variantou, avšak v pozdějších fázích přináší úspory nákladů díky konsolidovanému datovému modelu. U decentralizovaného uspořádání je tomu naopak. Problematice implementace DWH se věnuje např. [NOVOTNÝ, 2005], architekturami se zabývá např. [BÉBR, 2005]. Dostatek informací o DWH lze nalézt v [KIMBALL, 1998], [OMG, 2001b] popisuje kompletní metamodel DWH, o DWH jako integračním postupu píše např. [FOURNIER-MOREL, 2004]. V dalším textu budu používat obecný název DWH bez rozlišení jeho interní architektury.

3. Virtuální databáze - při vytváření jednotného přístupu ke všem datovým zdrojům je technologie závislá na konkrétním produktu. Používané technologie můžeme rozdělit podle jazyka pro přístup k datům:
    a. SQL (Simple Query Language) gateway - dotaz zadaný v SQL se transformuje a distribuuje do databází rovněž v SQL (připojených přes ODBC a JDBC).
    b. XQuery gateway - rozšiřuje možnosti SQL na dotazy do zdrojů s XML dokumenty, které mohou být připojeny např. jako webové služby.

17.3.2.2 Technologie funkčních rozhraní

1. MOM - nerozvinuto do samostatných technologií, známo jako technologie MOM.
2. MB - nerozvinuto do samostatných technologií, známo jako technologie MB.
3. ORB - integrační přístup ORB je implementován v konkurenčních technologiích:
    a. COM+/ DCOM (Distributed Common (někdy Component) Object Model)
    b. CORBA (Common Object Request Broker Architecture)
    c. Java/RMI (Remote Method Invocation) & JNDI (Java Naming and Directory Services)

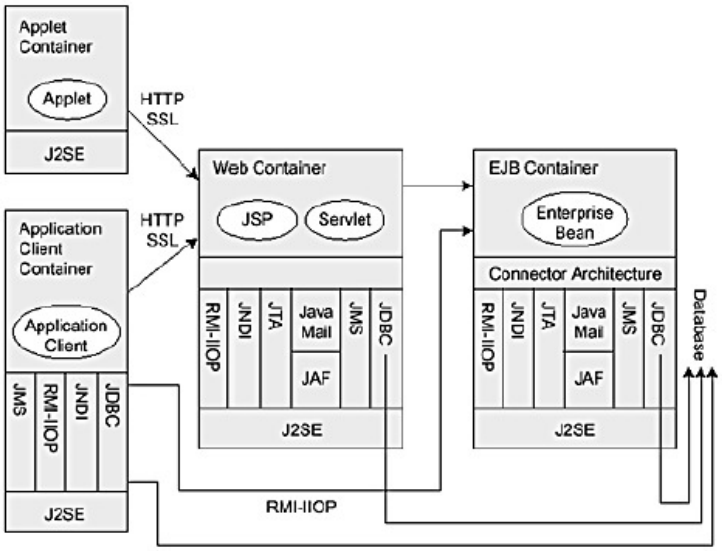
V tabulce 23 - Srovnání technologií ORB (zdroj: [SEI]) je jejich orientační srovnání.

ORB	Platforma	Aplikovatelnost	Mechanismus	Implementace
COM/DCOM	původní PC, rozšiřuje se na další	PC distribuovaná systémová architektura	API na proprietární systém	jedna
CORBA	nezávislá na platformě a spolupracuje mezi platformami	obecná distribuovaná systémová architektura	specifikace distribuované objektové technologie	mnoho
Java/RMI & JNDI	kdekoli na JVM (Java Virtual Machine)	obecná distribuovaná systémová architektura a webové intranety	implementace distribuované objektové technologie	různé

tabulka 23 - Srovnání technologií ORB (zdroj: [SEI])

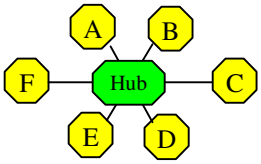
[TSE] nazývá tyto technologie souhrnně DOT (Distributed Object Technologies) a definuje je jako "typ middlewaru, který rozšiřuje koncepci objektově orientovaných technologií o distribuované zpracování. Rozhraní aplikací jsou vyvíjeny tak, aby vypadaly jako objekty".

4. SB - nerozvinuto do samostatných technologií, známo jako technologie SB. Pro komunikaci se službou se užívá SOAP (Simple Object Access Protocol), pro komunikaci s registrem služeb UDDI (Universal Description, Discovery and Integration Service).
5. TP - nerozvinuto do samostatných technologií, známo jako technologie TP.
6. AS - identifikujeme hlavní technologie AS podle architektury, kterou podporují - J2EE, .NET a CORBA.
   Aplikační servery v sobě zahrnují kompletní sadu technologií pro vývoj aplikací. Vedle technologií pro vývoj, resp. běh jádra aplikací (tedy implementaci obchodní logiky), jsou vývojářům k dispozici kompletní sady technologií pro tvorbu rozhraní aplikací. Nalezneme zde tedy všechny technologie zmíněné v kapitole 17.3.1 Přímá komunikace.
   Souhrn technologií pro tvorbu rozhraní je v J2EE nazýván J2CA (J2EE Connector Architecture). Tímto se detailně zabývá [SHARMA, 2001], kde se tak dočteme o možnostech integrace pomocí J2EE, stejně tak i v [SUN].



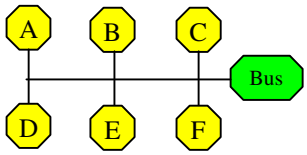
obrázek 65 - Přehled J2EE technologií (zdroj: [SHARMA, 2001])

7. BPI - podle topologie a způsobu řízení integrace rozlišujeme dvě technologie integračního přístupu BPI:
   a. Integrační hub (**Enterprise Integration Hub**) - aplikace spolu komunikují v topologii hvězdy, kde centrálním bodem je BPI, který integraci centrálně řídí.



obrázek 66 - Topologie integračního hubu (zdroj: [DIAMONDCLUSTER, 2001])

   b. Integrační sběrnice (v souvislosti s koncepcí SOA se používá termín **Enterprise Service Bus** (ESB), který zdůrazňuje orientaci na služby) - aplikace komunikují pomocí společné sběrnice, na kterou je připojen rovněž BPI. BPI dostává informace jako každá z připojených aplikací a řízení integrace je decentralizované přímo mezi aplikacemi.



obrázek 67 - Topologie integrační sběrnice (zdroj: [DIAMONDCLUSTER, 2001])

[KRISHNAN] definuje ESB jako "integrační stroj, který je řízen událostmi, podporuje distribuované operace a řízení, dále podporuje směrování a transformace dle obsahu a poskytuje platformu pro automatizaci a řízení podnikových procesů".
[MCCOY, 2003b] definuje ESB jako "aerodynamickou, distribuovanou integrační middleware infrastrukturu, která kombinuje podporu XML a webových služeb, základních transformací a obsahově založeného směrování. Buď obsahuje MOM nebo obaluje jiný transportní MOM mechanismus. Slouží jako odlehčená varianta integračního brokeru."
[PAPAZOGLOU, 2005] definuje ESB jako "otevřenou sběrnici pro zasílání zpráv, založenou na standardech, určenou pro implementaci, spuštění a řízení SOA řešení s důrazem na sestavování, spouštění a řízení distribuovaných servisně orientovaných architektur."
Mezi standardní schopnosti ESB patří dle [THOMAS]:

- událostmi řízený, dokumentově orientovaný procesní model,
- obsahově závislé směrování a filtrování,
- komplexní transformační schopnosti,
- podpora standardních rozhraní (COM, CISC, .NET, JMS, JCA, JDBC, Web Services),
- distribuované operace a management.

Detailně se popisu funkcionality ESB věnuje [CHAPPELL, 2004], který jej definuje jako "integrační platformu založenou na standardech, která kombinuje zasílání zpráv, webové služby, datové transformace a inteligentní směrování pro zajištění spolehlivého spojení a koordinace vztahů významného množství různých aplikací napříč podniky včetně transakční integrity". Rovněž uvádí tuto detailní charakteristiku ESB (přeloženo z [CHAPPELL, 2004]):

- "Všudypřítomnost - ESB je vhodný pro řešení potřeb obecných integračních projektů.
- Vysoce distribuovaná, událostmi řízená SOA - volně vázané integrační komponenty mohou být implementovány v rozsáhlých geografických topologiích a stále zůstávají dostupné jako sdílené služby na sběrnici.
- Výběrová implementace integračních komponent - adaptéry, transformační služby, obsahově založené směrovací služby mohou být implementovány podle potřeby (časově i místně) a nezávisle rozšiřovány.
- Bezpečnost a spolehlivost - všechny komponenty komunikující skrze sběrnici mohou využívat spolehlivé zasílání zpráv, transakční integrity a zabezpečené autentikace.
- Orchestrace a procesní tok - data protékají službami a aplikacemi bez ohledu na to, zda jsou lokální nebo vzdálené.
- Autonomní a federativně řízené prostředí - ESB podporuje lokální autonomii podnikových oddělení a přesto poskytuje prostředí pro integrované řízení.
- Přírůstková adaptace - ESB umožňuje postupovat formou menších projektů, které se postupně zapojují do celé integrované sítě.
- Podpora XML - XML je základním datovým typem.
- Sledování v reálném čase - ESB umožňuje sledování běhu podnikových dat v reálném čase prostřednictvím BAM komponenty."

V porovnání BPI na bázi integračního hubu a ESB spatřuje [CHAPPELL, 2004] dvě výhody ESB - ESB je schopen lépe podporovat distribuovaná podniková oddělení, neboť se jim blíží svou distribuovanou architekturou a autonomií komponent a za druhé ESB projekty jsou výrazně levnější, neboť používají standardní technologie na bázi webových služeb (to ovšem zároveň omezuje možnosti nasazení ESB na aplikace bez podpory webových služeb).
[GILPIN, 2004] definuje tři třídy ESB - odlehčený, standardní a high-end. Liší se podle funkcionality podle práce s metadaty, procesy, možností administrace a řízení až po integrační konektory. High-end ESB odpovídá plné funkcionalitě integračního serveru.
[JUŘEK, 2004] k tomuto rozdělení poznamenává : "Debaty o tom, zda je lepší používat uspořádání hubu nebo sběrnice jsou celkem zbytečné – v některých situacích se hodí to, v jiných ono. Většina integračních nástrojů je postavena spíše pro podporu uspořádání typu integrační sběrnice. Důvod je prostý – **sběrnice je obecnější koncept a vhodným nastavením podmínek lze v konkrétní situaci nastavit sběrnici tak, že funguje pro danou komunikaci jako hub**. Opačný postup, to je zobecnění funkce hubu pro princip sběrnice by byl určitě náročnější a mnohem méně intuitivní."

17.3.2.3 Technologie prezentačních rozhraní

1. Webový server - nerozvinuto do samostatných technologií, známo jako technologie webového serveru.
2. EIP - EIP bychom mohli rozdělit podle podporovaných technologií, které jsou uvedeny u integračního přístupu žádost o stránku. Není zpravidla možné navzájem kombinovat stránky generované odlišnými technologiemi (JSP, PHP, ASP apod.). Výsledná stránka se sestavuje pomocí tzv. **portletů** - umožňují používat pomocné značky (tzv. tagy) pro podporu integrace prezentační vrstvy, generování výsledné stránky probíhá na straně serveru.
3. Webový prohlížeč - nerozvinuto do samostatných technologií, známo jako technologie webového prohlížeče. Pro sestavení výsledné obrazovek z více zdrojů se používají tzv. **rámce** (frames) - jsou definovány ve standardu HTML; jejich variantou jsou tzv. inline frames (v definici standardu HTML). Detailní popis technologie rámců lze nalézt např. v [SHANNON, 2005]. Generování výsledné stránky probíhá na straně klienta podle instrukcí z hlavního rámce.
4. Snímání obrazovek - při tomto integračním přístupu se kombinuje více technologií dohromady. První skupinu tvoří **technologie pro komunikaci se zdrojovou aplikací**, např.
    - terminálová (telnet) a
    - webová (http)

   a druhou **technologie pro analýzu stránky** na základě:
    - zdrojového kódu stránky (např. HTML),
    - grafické rozpoznávání znaků (OCR - Optical Character Recognition),
    - rozeznávání objektů na obrazovce (např. OLE DB či ADO.NET technologie),

   Jejich spolupráci koordinuje konkrétní nástroj, který obsahuje i transformační pravidla a obchodní logiku pro práci se zdrojovou aplikací.

17.4 Integrační produkty

V této příloze si sestavíme přehled základních integračních produktů, které mají na trhu významnější zastoupení. Integrační produkty rozdělíme podle podporovaných integračních přístupů tak, jak byly popsány v tabulce 9 - Přehled integračních postupů (zdroj: autor). Cílem této kapitoly není srovnávat jednotlivé produkty (k tomu je zapotřebí detailní znalost srovnávaných produktů a především definice projektu, na jaký mají být aplikovány), ale poskytnout základní orientaci na trhu těchto produktů a získat konkrétní představu o tom, v kterých produktech se můžeme setkat s vybranými integračními technologiemi.

Zavádíme zde nový pojem - **třída produktů**. Tímto pojmem označuji skupinu produktů, které mají obdobnou funkcionalitu a zpravidla jsou v přehledech trhu uváděny společně. Tabulku 9 - Přehled integračních postupů (zdroj: autor) jsem nyní rozšířil o třídu produktů - viz tabulka 24 - Přehled integračních postupů rozšířený o třídy produktů (zdroj: autor), která spojuje integrační technologie do množin tak, jak jsou dostupné v integračních produktech.

Poznámky k dalšímu textu v této kapitole:

- Třídy produktů jsou uměle vytvořené a nemusí souhlasit s marketingovými kategoriemi výrobců - slouží pouze pro udržení přehlednosti v členění.
- Některé technologie nalezneme ve více třídách produktů - např. MOM je ve třídě produktů specializovaných pouze na MOM a zároveň ve třídě integračních serverů, jejichž část funkcionality pokrývá rovněž MOM.
- Pokud je u popisu produktů uvedeno umístění na trhu, je tak uvedeno pouze jako doplňující informace, která může být zkreslená podle informačního zdroje a době jeho publikace.

Třída produktů	**Technologie přímé komunikace**	**Přístupy přímé komunikace**	**Kategorie rozhraní**	**Přístupy zprostředkované komunikace**	**Technologie zprostředkované komunikace**	**Třída produktů**
Datové						
OS + nadstavby	Souborové	Replikace		Replikace a transformace	ETL	ETL
					Transformace na zakázku	DB, OS
DB	Databázové			Datový sklad	ODS	DB
					DWH	DWH
DB	Databázový link	Databázový link		Virtuální databáze	SQL gateway	SQL gateway
					XQuery gateway	XQuery gateway
Funkční						
AS (CORBA)	CORBA RPC	RPC		ORB	COM+/DCOM	AS (COM+/DCOM)
AS (J2EE)	Java RMI, EJB				CORBA	AS (CORBA)
AS (.NET)	.NET Remoting				Java/RMI & JNDI	AS (J2EE)
DB	PL/SQL procedura					

AS	Webová služba			SB	SB	AS, BPI, ESB
AS (CORBA)	CORBA Messaging	Zasílání zpráv		MOM	MOM	MOM, BPI
AS (J2EE)	JMS					
AS (.NET)	MSMQ			MB	MB	MB, BPI
AS	Webová služba					
AS (CORBA)	CORBA Transaction	Konverzační		TP	TP	TP, BPI
AS (J2EE)	JTS			AS	založený na .NET	AS (.NET)
					založený na J2EE	AS (J2EE)
					založený na CORBA	AS (CORBA)
AS (.NET)	.NET Transaction			BPI	Integrační hub	BPI
					Integrační sběrnice	BPI, ESB
Prezentační grafický						
EIP (.NET)	ASP	Žádost o stránku		Webový server	Webový server	Webový server
EIP (.J2EE)	JSP			EIP	EIP (portlety)	EIP
Webový server + PHP plugin	PHP			Webový prohlížeč	Webový prohlížeč (rámce)	Webový prohlížeč
Prezentační obalový						
n/a	n/a	n/a		Snímání obrazovek	analýza zdrojového kódu	PrIS
					OCR	
					rozeznávání objektů	

tabulka 24 - Přehled integračních postupů rozšířený o třídy produktů (zdroj: autor)

17.4.1 Operační systém (OS + nadstavby)

Produktům třídy OS se zde nebudeme podrobněji věnovat, neboť z pohledu integrace nás zajímá pouze schopnost kopírovat či přesunovat (a případně drobně upravovat) soubory s daty, což je z pohledu OS elementární funkcionalita. Nadstavbou zde míním funkcionalitu doplňující OS o alternativní způsoby přenosu souborů, např. pomocí protokolu FTP. Těchto nadstaveb je dostupná celá řada a není třeba uvádět konkrétní příklady.

V oblasti operačních systémů dominují OS na bázi unixu a MS Windows. Pro přenos souborů se také využívá funkcionalita JVM (Java Virtual Machine), kterou bychom mohli označit za nadstavbu OS.

17.4.2 Databáze (DB)

V oblasti databází, resp. databázových systémů dominují tři hlavní dodavatelé dle [LAMONICA, b]) - Oracle, IBM, Microsoft. O některých dalších produktech se zmiňuje např. [NOVOTNÝ, 2005] a velmi užitečný průvodce [FRENTZEN, 2003].

17.4.3 Podnikové informační portály (EIP)

Platforma pro vybudování podnikového informačního portálu se nazývá portálový framework. Portálový framework je zpravidla nadstavbou aplikačního serveru a rozšiřuje jeho technologie o podporu prezentační vrstvy. Mezi přední dodavatele portálových frameworků dle [PHIFER, 2004] patří IBM, SAP, BEA Systems, Oracle, Plumtree Software, Sun Microsystems, Peoplesoft a Vignette. Podobný přehled nabízí i [RAMOS, 2003a] a [MURPHY, 2002].

17.4.4 ETL

Mezi dodavatele produktů ETL patří dle [FRIEDMAN, 2003] např. Informatica, Acsential, Business Objects a Oracle. Podobný přehled je rovněž v [NOVOTNÝ, 2005].

17.4.5 Datawarehouse (DWH)

Pro vybudování DWH se používají 4 třídy produktů (viz [NOVOTNÝ, 2005]):

- databáze - viz kapitola 17.4.2 Databáze (DB),
- ETL - viz kapitola 17.4.4 ETL,
- pokročilé analytické aplikace a
- nástroje pro data mining.

K posledním dvěma odrážkám nebudeme uvádět dodavatele, neboť tyto třídy produktů nemají přímý vztah k integraci. Zájemce odkazuji na [NOVOTNÝ, 2005].

17.4.6 SQL Gateway

[LINTHICUM, 2003] a [LINTHICUM, 1999] uvádí příklady těchto produktů SQL Gateway - Information Builders' Enterprise Data Access/SQL (EDA/SQL), IBM's Distributed Relational Data Access (DRDA) a ISO/SAG's Remote Data Access (RDA). Tyto produkty jsou patrně postupně vytlačovány konkurenčními produkty XQuery Gateway, neboť nabízejí širší funkcionalitu.

17.4.7 XQuery Gateway

Tyto produkty jsou založeny na standardu XML Query, který se dosud dopracovává [W3C, 2004], ale který je principiálně zaměřen na práci s daty z více různorodých zdrojů. Rovněž trh s těmito produkty se teprve rozvíjí, přesto můžeme jmenovat následující nástroje, které jsou již na trhu, nebo se na něm v brzké době objeví (dle [FOURNIER-MOREL, 2004]) - IBM (DB2 Information Integrator (alias Xperanto)), BEA (LiquidData), Oracle (9iDB R2 (XML DB)), BlueStream (XStreamDB), Microsoft (SQL Server ("Yukon")), eXcelon Corp. (XIS), X-Hive (X-Hive/DB), Fatdog (XQEngine), SAGENT (OpenLink), Actuate (NIS (alias Nimble)), Vignette (BIStudio), Day (Communiqué), Documentum (Content Aggregation Services), Callixa (CIS), Venetica (VeniceBridge), ContextMedia (InterChange) aQexo (a GNU open-source). Detailním přehledem a srovnání těchto produktů se zabývá [KERNOCHAN, 2003].

17.4.8 MOM

Produkty MOM se za dobu své existence rozvinuly do komplexních produktů BPI, [LINTHICUM, 2003] se zmiňuje dva příklady MOM - IBM MQSeries a Microsoft MSMQ.

17.4.9 Message broker (MB)

Produkty MB vznikaly z produktů MOM přidáním funkcionality zprostředkovatele a rovněž se rozvinuly až do BPI. Z dodavatelů MB produktů (také dle [LINTHICUM, 1999]) jmenujme IBM (MQ Integrator), Tibco (Rendezvous), Active Software, Neon, TSI Software, SAGA Software.

17.4.10 Transakční monitor (TP)

Transakční monitory se rozvinuly z produktů MB přidáním podpory transakcí a nyní jsou rovněž rozvinuty až do BPI. Z [LINTHICUM, 2003] vybírám příklady produktů TP - Tuxedo (BEA Systems), Microsoft Transaction Server (Microsoft) a CICS (IBM).

17.4.11 Aplikační server (AS)

Aplikační servery, jak již bylo řečeno, jsou aplikačním frameworkem pro tvorbu aplikací (primárně webových) a z toho důvodu musí obsahovat i všechny potřebné technologie pro tvorbu integrovaných aplikací přes funkční rozhraní. Existují tři konkurenční množiny těchto technologií - .NET, J2EE a CORBA, které navzájem nejsou příliš kompatibilní. CORBA je považována za méně perspektivní, proto se s ní setkáváme spíše u starších aplikací. COM+/DCOM je předchůdcem .NET, a proto jej zde explicitně neuvádím. Uveďme si příklady produktů AS:

1. .NET - .NET Framework (nadstavba k OS MS Windows).
2. J2EE - AS Weblogic (BEA Systems), WebSphere (IBM), 10g AS (Oracle), SunOne (Sun Microsystems), AS Borland, Orbix E2A (Iona), NetWeaver Web AS (SAP), Enterprise AS (Sybase), jBoss AS (dle IDC [KRILL, 2004]).
3. CORBA - rozsáhlý přehled viz [CORBA].

Společný přehled trhu s J2EE a .NET aplikačními servery poskytuje např. [NATIS, 2003a].

V souvislosti s rozvojem webových služeb, resp. SODA vznikají i specifické nástroje (oproti standardním vývojářským prostředím) pro vývoj webových služeb. Tyto nástroje se nazývají **ISE** (Integrated Service Enviroment) a měly by podporovat tvorbu grafického designu GUI, tvorbu portletů, UML, repository metadat, IDE, legacy adaptéry, grafické modelování služeb a jejich vazeb, **modelování procesů (workflow)**, tvorbu spouštěcích mechanismů pro procesy (workflow), kontroly konzistence a registrace služeb a předání do provozu. Podle [PLUMMER, 2003b] tyto produkty dodává např. BEA Systems, IBM, Microsoft, Novell a Bowstreet.

17.4.12 Integrační server (BPI)

Integrační servery tvoří jeden z vrcholů evoluce současných nástrojů pro integraci aplikací. Integrují do sebe několik různých technologií - MOM, MB, SB, TP a AS. Samozřejmě to platí pouze v ideálním případě - existuje řada produktů, které se zahrnují rovněž do skupiny BPI, ačkoli např. neobsahují service broker či jinou skupinu technologií. Za produkty třídy integračního serveru považujeme produkty, které podporují koncepci EAI, tedy oproti jiným podporují modelování a běh podnikových procesů při integraci aplikací.

Přehled trhu BPI nabízí několik informačních zdrojů - např. [GROENENDAAL, 2002], [TEILHARD, 2003], [PAC, 2003], [MCCOY, 2003a] a [VOLLMER, 2005], podle kterých mezi přední dodavatele BPI patří IBM, Tibco, webMethods, SeeBeyond, Microsoft, BEA Systems, ORACLE a Sybase.

17.4.13 Enterprise Service Bus

Produkty ESB označuje [MCCOY, 2003b] za "odlehčenou" verzi integračních serverů. Jsou zaměřené na webové služby a SOA (veškerá komunikace probíhá pomocí webových služeb a jejich standardů), komunikace probíhá v topologii sběrnice. Oproti plnohodnotným integračním serverům zpravidla postrádají nástroje na modelování procesů, adaptéry,

vývojářské nástroje a nástroje pro administraci a monitorování. Jejich cena tvoří cca 10 procent ceny plnohodnotných EAI nástrojů (uvádí [SCHULTE, 2003]). Z tohoto důvodu bývají označovány jako low-end integrační brokery.

Příklady produktů ESB uvádí [SCHULTE, 2004] - např. ESB (Fiorano Software), Artix (IONA Technologies), Jintegrator (PolarLake), Insight ESB (SeeBeyond), ESB (Sonic Software) a 4 Server (Cape Clear Software). V roce 2005 se do produktů ESB začali přesunovat i dodavatelé BPI, konkrétně např. BEA (AquaLogic), Tibco (ESB), IBM (WebSphere ESB). Rovněž tak se připravuje Microsoft s produktem Indigo.

Společnost Gartner také používá příbuzné označení "podnikový nervový systém" (**ENS** - Enterprise Nervous System) pro produkty určené pro podporu "SOA, událostmi řízených aplikací a různých dalších designových vzorů pro kompozitní aplikace, synchronizaci dat v reálném čase a vícekrokové podnikové procesy" (převzato z [SCHULTE, 2005]). Tato skupina produktů zahrnuje produktové kategorie ESB, integrační balíky (BPI) a APS. Mezi dodavatele této třídy produktů patří dle [SCHULTE, 2005] např. Microsoft, IBM, Tibco, SeeBeyond, webMethods, Fujitsu, Oracle a BEA Systems.

17.4.14 Webový server

V oblasti open-source webových serverů je patrně nejvíce rozšířen produkt Apache. Vedle něj existuje řada komerčních webových serverů, které jsou dodávány jako součást celých řešení, a nemá význam je jmenovat jednotlivě. K webovým serverům je možné nainstalovat dodatečné moduly - jedním z nich je i podpora PHP.

17.4.15 Webový prohlížeč

Mezi nejpoužívanější webové prohlížeče patří MS Internet Explorer, Opera, Netscape Communicator a Firefox.

17.4.16 Programmatic Integration Server (PrIS)

S nástupem webových služeb společně se SOA koncepcí se na trhu objevila i další skupina produktů, které umožňují obalit legacy aplikace, a vytvořit tak integrační rozhraní k těmto jinak nepřístupným aplikacím. Tyto produkty se nazývají PrIS a [VECCHIO, 2003a] poskytuje přehled jejich dodavatelů - mezi nimi např. IBM, Jacada, ClientSoft, WRQ, Seagul Software Systems a Neon Systems.

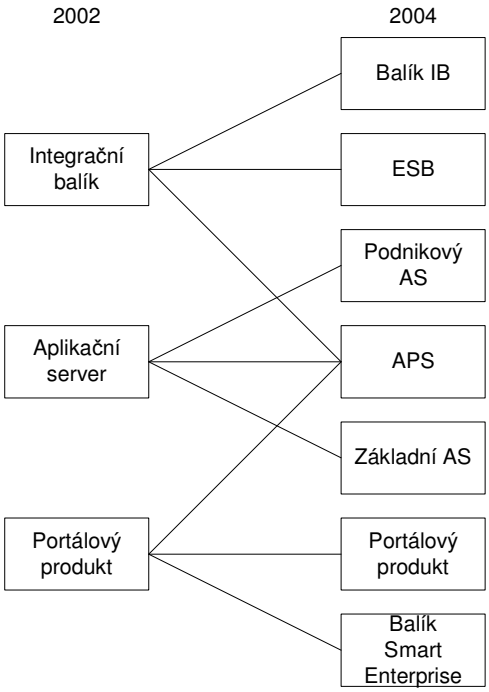
Vedle produktů PrIS existují i produkty specializované na tzv. **web clipping** (emulující klikání uživatele ve webové aplikaci), které jsou určené pro obalování webové prezentační vrstvy a využívají protokolu HTTP. Příkladem takového produktu je Kapow RoboSuite.

17.4.17 Aplikační platforma (APS - Application Platform Suite)

Významní dodavatelé v oblasti integrace poskytují současně produkty v několika třídách. Jejich kombinací vznikají kompletní sady pro podniková řešení. [NATIS, 2002] takové řešení nazývá aplikační platformou a měla by obsahovat následující komponenty:

1. podnikový aplikační server (AS),
2. podnikový portál (EIP) a
3. podnikovou integrační platformu (BPI).

Provázanost těchto produktů je graficky znázorněna na obrázku 68 - Vztah APS a AS, EIP, BPI (zdroj: [NATIS, 2003a]). [DONNELLY, 2004] uvádí přehled dodavatelů APS - např. BEA Systems, IBM, Oracle, Sun, Novell a Microsoft. Výhodami APS se zabývá [GARTNER, 2003].



obrázek 68 - Vztah APS a AS, EIP, BPI (zdroj: [NATIS, 2003a])

17.5 Kritéria výběru integračního serveru

Přehled kritérií pro výběr integračního serveru vznikl provázáním kritérií z [OVUM, 2001] na model integračního serveru v kapitole 12.1 Integrační server, jejich aktualizací a doplněním autorem a rozdělením dle úrovní integrace.

Kritéria			**Úroveň integrace (Datová, Služeb, Procesní)**
úroveň 1	úroveň 2	úroveň 3	
Identifikace produktu	Název		DSP
	Číslo verze		DSP
	Výrobce		DSP
	Detaily o výrobci		DSP
	Základní poslání produktu		DSP
	Reference		DSP
Architektura	Aplikační a technologické adaptéry	Adaptéry založeny na metadatech	SP
		Podpora transformace mezi protokoly	SP
		Podpora více vláken v jednom adaptéru	DSP
		Podpora neinvazivního přístupu v adaptérech	SP
		Podporované protokoly	DSP
		Podporované technologie rozhraní	DSP
		Předpřipravené aplikační adaptéry	DSP
	Message broker	Podpora různých komunikačních modelů	DSP
		Podpora serializace zpráv	DSP
		Podpora nastavení priorit zpráv	DSP
		Podpora parametrů vypršení zprávy	DSP
		Podpora garantovaného doručení	DSP
		Podpora synchronního i asynchronního zpracování	DSP
		Nezávislost na protokolu	DSP
		Odolnost fronty zpráv proti výpadku	DSP
		Inteligentní směrování s kontrolními body	DSP
		Podporované formáty zpráv a možnost využití dalších pluginů	SP
		Obsahově orientované směrování	DSP
	Service broker	Podpora automatické registrace služeb	SP
		Obsah repository registrovaných služeb	SP
		Podpora webových služeb	SP
	Řízení událostí	Událostmi řízené prostředí	SP
		Nástroj pro monitorování a řízení událostí	SP
	Business rules engine	Interní nebo externí BRE	DSP
		Nástroje pro udržování repository business pravidel	DSP
		Možnosti definování pravidel	DSP
	Transakční monitor	Podpora transakčního zpracování	SP
		Podpora roll-back transakce	SP
		Podpora transakcí typu XA	SP

	Repository metadat	Repository standardních zpráv	SP
		Vícejazyčná podpora	SP
		Regionální podpora	SP
		Podpora vazeb mezi datovými elementy	SP
		Podpora exportu/importu metadat	SP
	BAM	Podpora měření KPI	P
		Nástroj pro monitorování aktivit	P
	BPM	Manažer stavů procesů	P
		Řešení chybných procesů	P
		Řešení nedostupných aplikací	P
		Monitorování stavu a historie procesů	P
	Zabezpečení	Session manager	DSP
		Podpora uživatelských profilů	SP
		Podpora propagace uživatelů procesů	P
		Auditování činností uživatelů procesů, systému	SP
		Podpora delegované administrace	DSP
		Podpora digitálních certifikátů	SP
		Podpora bezpečnostních standardů	DSP
		Komunikace založena na autentizaci, autorizaci, privacy, zachování integrity, nepopiratelnosti	DSP
	Otevřenost	Otevřené API pro dokument management	DSP
		Otevřené API pro content management	DSP
		Otevřené API pro workflow produkt	P
		Otevřené API pro OLAP	DSP
		Otevřené API pro reportovací produkt	DSP
		Otevřené API pro produkt řízení bezpečnosti	DSP
Vývoj	Vývojové nástroje	Podporované programovací jazyky	DSP
		Intuitivní GUI nástrojů	DSP
		Snadnost použití vývojových nástrojů	DSP
		Úroveň integrace mezi vývojovými nástroji	DSP
		Lokalizace nástrojů	DSP
		Podpora verzování během vývoje	DSP
		Nástroj pro modelování procesů	P
		Nástroj pro vývoj adaptérů	DSP
		Nástroj pro analýzu závislostí a dopadů uvnitř systému	SP
		Podpora RAD, JAD	DSP
		Podpora testování	DSP
		Podpora týmové spolupráce	DSP
		Podporované platformy pro adaptéry	DSP
		Podporované platformy pro transformační engine	SP
		Podporované platformy pro administraci	DSP
		Podporované platformy pro procesní engine	P
		Drag and drop podpora mapování	DSP
		Podpora podmíněných transformací	DSP
		Podpora externích transformačních algoritmů	DSP
	Normalizace	Podporované standardy	SP
		Znovupoužitelnost nově vytvořených transformačních algoritmů	SP
		Normalizace vytvářených společných prvků	SP

		Úroveň automatického generování dokumentace	DSP
		Úroveň generování programového kódu z nástroje pro modelování procesů	P
		Předpřipravené transformační algoritmy	SP
		Předpřipravené adaptéry pro aplikace	DSP
		Předpřipravené adaptéry pro databázové systémy	DSP
		Předpřipravené adaptéry pro aplikační servery	DSP
		Předpřipravené standardní podnikové procesy	P
	Modelování procesů	Podpora rozhodovacích procesů	P
		Podpora konkurentních procesů a komunikace mezi nimi	P
		Podpora vnořených procesů	P
		Podpora externích procedur	P
	Testovací nástroje	Generátor zpráv	DSP
		Sběrač dat	DSP
		Analyzátor logů	DSP
		Reaktor simulující činnost adaptérů	DSP
		Debuggovací nástroj pro běh procesů	P
Zavádění a provoz	Administrace	Podpora verzování během nasazování	DSP
		Podpora automatického nastartování	DSP
		Podpora restartu jednotlivých komponent řešení	DSP
		Podpora podmíněného deploymentu	SP
		Podpora automatizovaného řízení vyjímečných stavů	DSP
		Zasílání zpráv o stavu systému	DSP
		Distribuovaná administrace	DSP
		Podpora rekonfigurace systému a přidávání/odebírání komponent za provozu	DSP
		Change management	DSP
		Podpora automatického zálohování	DSP
	Monitorování	Diagnostika systému s intuitivním GUI	DSP
		Reportovací nástroj	DSP
		Detekce úzkých míst	DSP
	Výkonnost	Podpora přerozdělování výkonu	DSP
		Optimalizace využití výkonu	DSP
		Škálovatelnost objemu zpráv	DSP
		Škálovatelnost adaptérů	DSP
		Zabezpečení proti výpadku, nejčastěji formou replikace klíčových komponent	DSP
		Propustnost	DSP
		Zpoždění	DSP
		Počet konkurentních transakcí/zpráv	DSP
Rozšiřitelnost	Software	Otevřenost produktu pro instalaci pluginů třetích stran	DSP
		Množina doplňkových produktů třetích stran	DSP
		Předpokládaný rozvoj produktu	DSP
Cena	Licenční politika		DSP
	Provozní náklady		DSP
	Roční support/update		DSP

	Školení		DSP
	Licence produktů třetích stran		DSP
	Náklady na upgrade		DSP

tabulka 25 - Kritéria výběru integračního serveru (zdroj: autor a [OVUM, 2001])

17.6 Případové studie

V této příloze stručně představím integrační projekty, na kterých jsem se podílel v různých rolích - od business analytika, architekta integrace, architekta datových struktur, architekta podnikových procesů, specialisty na informační bezpečnost, manažera testů až po testera dodavatele. Z poznatků z těchto projektů jsem vycházel při psaní této práce a zároveň jsem si na nich ověřoval své náměty a postupy. Na projektech jsem spolupracoval s řadou expertů, kteří se těmto projektů věnují velkou část svého profesního života, a oni byli rovněž velkými dárci znalostí, které jsem při tvorbě této práce uplatnil. Při rozhodování o použité metodice, či integrační technologii jsem neměl v žádném z těchto projektů rozhodující slovo. Přesto jsem měl možnost prosazovat své návrhy popsané v této práci, přebírat zodpovědnost za jejich úspěšnou aplikaci a tím i ověřovat jejich správnost.

O nutnosti přizpůsobit metodiku konkrétnímu projektu píše [VOŘÍŠEK, 1997] i [BUCHALCEOVÁ, 2005]. Metodika prezentovaná v této práci je přizpůsobená vybrané skupině projektů a šetří tak určitou část nákladů nutných pro přizpůsobování konkrétnímu projektu z této skupiny. [BUCHALCEOVÁ, 2005] identifikuje 15 atributů projektu. Podle poznatků z reálných projektů, které jsou vzorkem pro české prostředí, jsem prostřednictvím těchto atributů provedl společnou charakteristiku těchto integračních projektů v tabulce 26 - Atributy zkoumané skupiny integračních projektů podle MeFIS (zdroj: autor).

Atribut	Popis	Hodnota
A01	Problémová doména	EAI
A02	Typ řešení	Jedna z uvedených možností nebo kombinace: NEW (vývoj nového řešení), INT (integrace řešení), UPG (rozvoj a rozšíření řešení).
A03	Způsob řešení	IN (interní) i OUT (externí).
A04	Přístup k řešení	Nelze obecně říci, podle integračního přístupu.
A05	Důležitost systému	Většinou H (mission critical) - pro podporu hlavních podnikových procesů.
A06	Strategická příležitost	Nejvíce odpovídá kategoriím Rozběh (vše pro nasazení řešení) a Udržení pozic.
A07	Počet lidí v týmu	Nelze obecně říci, záleží na počtu integrovaných aplikací a rozsahu implementované funkcionality; na projektech, na kterých jsem se podílel, se tým při NEW a INT pohyboval v kategoriích 50 - 100, při UPG 10 - 20 aktivních členů.
A08	Rozmístění lidí v týmu	Nelze obecně říci.
A09	Kooperace zákazníka	Zpravidla nutná denní spolupráce a konzultace.
A10	Specifické požadavky	Spolehlivost, rychlost, ostatní nelze obecně říci.
A11	Priority projektu	Bezchybovost, ostatní nelze obecně říci.
A12	Kvalifikace lidí v týmu	Podle rolí v týmu, zejména znalost podnikových procesů a integrovaných podnikových aplikací na straně zákazníka; znalost integrační platformy a integračních postupů na straně dodavatele.
A13	Vlastnosti lidí v týmu	Nelze obecně říci.
A14	Pracovní prostředí	Sdílení dokumentů, ostatní nelze obecně říci.
A15	Vývojové nástroje	Nástroje pro řízení projektu (např. MS Project), pro dokumentaci analýzy, návrhu a testování (např. MS

		Word), pro návrh, vývoj a kustomizaci integračních komponent (specifické nástroje podle integračních produktů včetně CASE nástrojů), pro verzování zdrojových kódů (např. CVS, PVCS Version Manager), pro správu souborů (např. MS Explorer), pro editaci XML souborů (např. XML Spy), pro práci s databází (např. TOAD), pro testování (proprietární nástroje podle zvolených integračních postupů), pro reportování chyb (např. PVCS Tracker), integrační produkty.

tabulka 26 - Atributy zkoumané skupiny integračních projektů podle MeFIS (zdroj: autor)

17.6.1 Projekt 1 - Podnikový informační portál (2000 - 2003)

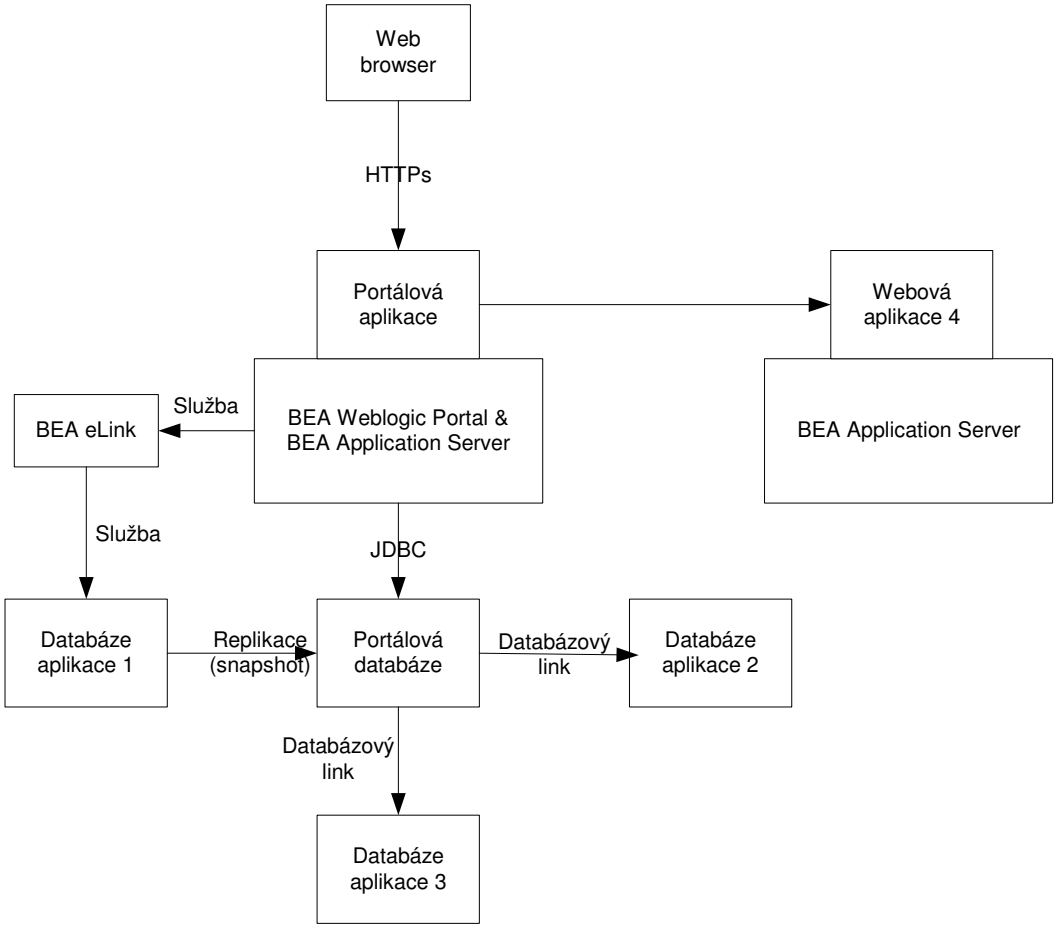
Cílem projektu bylo vybudovat zákaznicky orientovaný portál, který prezentuje vybraná zákaznická a produktová data prostřednictvím webového prohlížeče zaměstnancům i zákazníkům. Necháme-li stranou veškerou portálovou funkcionalitu, která nemá s integrací přímo nic společného, pak projekt prošel třemi stádii:

1. Vybudování základního portálového frameworku, do kterého bylo zaintegrováno cca 120 portletů generovaných lokálně. Tyto portlety čerpaly data z portálové databáze, do které byla replikována data o produktových instancích a přes databázové linky se přistupovalo k číselníkům dalších 2 aplikací. Tato fáze trvala cca rok.
2. Funkce portálové aplikace byla rozšířena o možnost zapisovat data od zákazníků do interních aplikací. Tato funkcionalita musela být řešena z bezpečnostních důvodů přes integrační broker, který změny propagoval do cílové aplikace. Takto byly realizovány 2 integrační případy, přesto toto rozšíření funkcionality zabralo skoro půl roku.
3. Vedle portálové aplikace začal podnik vyvíjet další webové aplikace, které nebyly přímo integrovány do portálu, ale vzdáleně poskytovaly přes proprietární protokol služby, které poskytovaly předzpracovaná data pro portlety. Tím se umožnilo distribuovat zátěž způsobenou během aplikační vrstvy aplikací na více aplikačních serverů.

Během tohoto projektu jsem neměl prakticky žádné know-how o integraci aplikací, a tak jsem zde pouze čerpal poznatky. Při vývoji aplikace byla užita obecně zaměřená firemní metodika. V porovnání s pozdějšími projekty bylo řešení integračních úloh velmi zdlouhavé a často měněné. Příčinou byly chybějící postupy pro analýzu a zejména návrh integračních případů a rozsáhlé diskuse se zákazníkem o volbě vhodné integrační technologie.

Ve druhé fázi projektu se seriózně uvažovalo o přepracování všech přístupů do aplikací do formy služeb integračního brokeru. Počet těchto služeb se pohyboval mezi 40 - 120 (opět nejednotný přístup k integračním případům) a odhadované náklady se blížily celkové sumě investic do stávajícího řešení. Jediným důvodem pro tuto změnu byl požadavek zákazníka na zvýšení bezpečnosti interních aplikací. Tento návrh byl po dlouhých diskusích zamítnut.

Integrační schéma z projektu 1 je na obrázku 69 - Integrační schéma projektu 1 (zdroj: autor).



obrázek 69 - Integrační schéma projektu 1 (zdroj: autor)

Ověřené závěry z projektu pro tuto práci:

- Podnikové portály jsou integračním nástrojem prezentační vrstvy. Popis integračních případů prezentační vrstvy zahrnuje datové zdroje, jejich formátování a napojení na portálový framework. Integrační proces prezentační vrstvy je zpravidla iniciován uživatelskou akcí na prezentační vrstvě a ukončen aktualizací prezentační vrstvy.
- Podnikové portály je možné integrovat prostřednictvím datových rozhraní, synchronních služeb integračního brokeru a prezentačních rozhraní jiných aplikací. Data pro prezentační vrstvu jsou vyžadována on-line.
- Ve vztahu ke komplexnímu modelu integrovaného PIS byly ověřeny integrační cesty skrze EIP, integrační server a částečně ODS (portálová databáze se obsahem ODS blíží).
- Referenční model EIP byl ověřen analýzou funkcionality portálových produktů, která předcházela výběru produktu BEA WebLogic Portal.
- Analýza bezpečnostních rizik řešení (dimenze SEC) je podstatnou součástí integračního projektu, neboť každé rozhraní činí podnikovou aplikaci otevřenější vůči potenciálnímu útoku. Zvláště pak u řešení typu podnikového portálu, kdy je systém otevřený vně podniku.
- Nejasná integrační strategie a rozhodování o vhodném integračním postupu vedly k nadbytečné spotřebě času a nákladů na projekt.

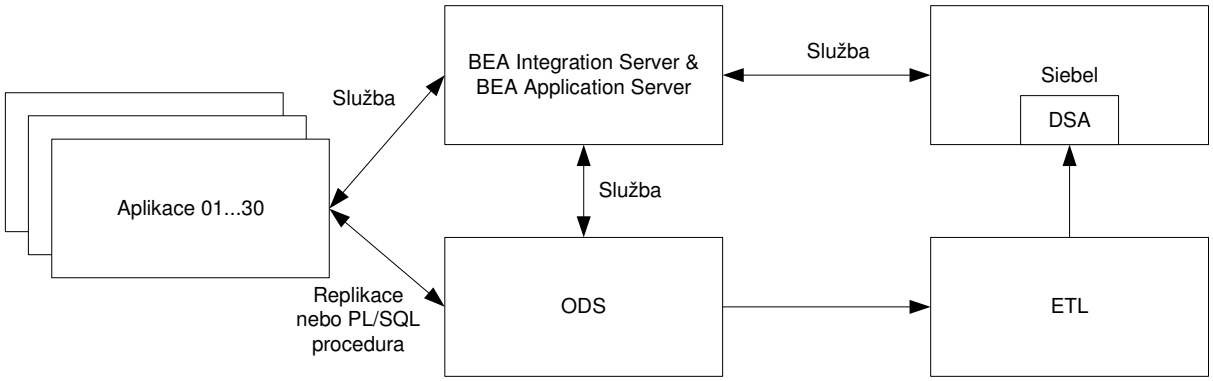
Tyto závěry jsou podloženy projektovou dokumentací [CSC, 2003] (technickou specifikací, studií na téma Porovnání produktů pro realizaci podnikových informačních portálů, analýzou bezpečnostních rizik aplikace (CRAMM)), která ovšem není veřejná, neboť je majetkem zákazníka a je chráněna obchodním tajemstvím.

17.6.2 Projekt 2 - Integrace CRM (2004)

Cílem druhého integračního projektu byla výměna CRM v podnikovém informačním systému. V tomto projektu bylo cílem integračního týmu navrhnout a implementovat ODS s produktovými a zákaznickými informacemi, navázat se na podnikové procesy implementované v CRM a protáhnout je do ostatních podnikových aplikací prostřednictvím integračního serveru, zajistit přelévání velkých objemů dat do ODS a odtud do CRM a v neposlední řadě navrhnout postup pro odstavení starého CRM a připojení nového CRM s minimálním dopadem na podnik.

Počet integrovaných aplikací se pohyboval kolem 30 a během analýzy bylo postupně detekováno dalších deset aplikací, na které se při zadání projektu pozapomnělo.

Integrační schéma projektu je znázorněno na obrázku 70 - Integrační schéma projektu 2 (zdroj: autor).



obrázek 70 - Integrační schéma projektu 2 (zdroj: autor)

V tomto projektu jsem spolupracoval s několika integračními experty, kteří díky zažitému know-how dokázali postupně identifikovat integrační procesy i integrační případy a detekovat jejich vzájemné závislosti. Na konci analýzy bylo popsáno cca 80 integračních procesů. Výsledkem tohoto strukturovaného a metodického přístupu byla změna celého přístupu k implementaci CRM (kde se od začátku předpokládalo postupovat podle jednotlivých produktů, avšak jelikož integrační případy je nutno řešit současně pro všechny produkty a naopak je efektivnější postupovat podle jednotlivých aplikací), navýšení rozpočtu projektu o cca 150 procent (zákazník v původních odhadech podcenil náročnost požadované integrace) a v konečném důsledku zastavení projektu po dokončení detailní fáze analýzy (kdy bylo možné na integračních případech jednoznačně dokladovat dopad dodatečných zákaznických požadavků nad původní rámec projektu).

Ověřené závěry z projektu pro tuto práci:

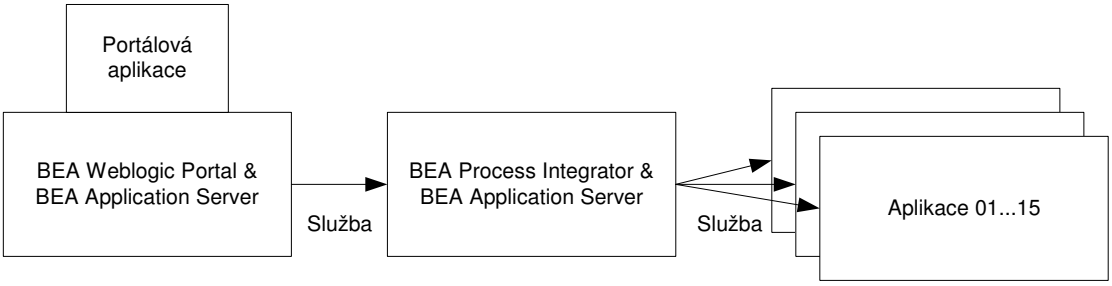
- Použití abstrakce integračního případu a procesu pro analytickou fázi projektu, což vedlo k výraznému zprůhlednění a porozumění projektu.
- Byly ověřeny integrační cesty skrze integrační server, ETL, DSA a ODS a větve rozhodovacího stromu pro výběr mezi těmito integračními postupy.
- Aplikace integrační strategie pro integraci podnikových aplikací skrze ODS, integrační server a ETL.
- Referenční model BPI byl ověřen na produktu BEA WebLogic Integrator.

Tyto závěry jsou podloženy projektovou dokumentací [NESS, 2004] (analytickým výstupem, integrační koncepcí), která ovšem není veřejná, neboť je majetkem zákazníka a je chráněna obchodním tajemstvím.

17.6.3 Projekt 3 - Aplikace pro zřizování služby (2004, 2006)

Ve třetím projektu šlo o vytvoření webové aplikace v portálovém frameworku, přes kterou si zákazník mohl objednat zřízení služby. Tento požadavek byl předán do integrační platformy k vyřízení, čímž se rozeběhl složitý integrační proces, který v několika aplikacích ověřoval, zda si zákazník může službu zřídit a zda mu ji podnik může ze svých zdrojů poskytnout. Dále objednávku zaevidoval a postupně předával mezi jednotlivými aplikacemi integračním procesem tak, že sledoval její stav a směroval ji k vyřízení. Tento systém je schopen současně řídit několik tisíců objednávek.

Integrační schéma je na obrázku 71 - Integrační schéma projektu 3 (zdroj: autor).



obrázek 71 - Integrační schéma projektu 3 (zdroj: autor)

Tento projekt dopadl úspěšně, řešil cca 15 složitých integračních procesů. Do této práce jsem z něj přenesl poznatky právě z rozsáhlých a dlouhotrvajících integračních procesů.

I můj další integrační projekt byl prakticky identický, ačkoli iniciátorem procesů nebyla portálová aplikace, ale webová služba dostupná externímu partnerovi podniku. Z hlediska integrace se však jednalo o obsahově shodný projekt, proto ho zde explicitně neuvádím.

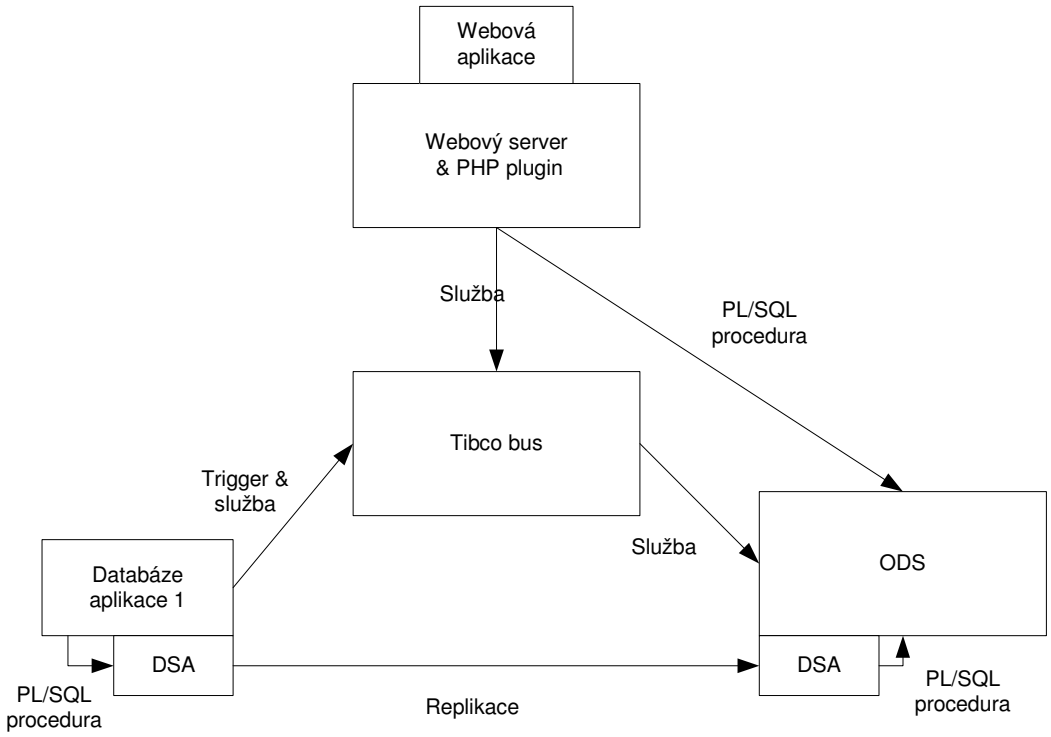
Ověřené závěry z projektu pro tuto práci:

- Použití abstrakce integračního případu a procesu pro komplexní integrační úlohy.
- Byly ověřeny integrační cesty skrze integrační server.
- Přechod od integračního případu k návrhu jeho řešení integračním postupem a následně detailním návrhem implementace na integrační platformě.
- Přechod od integračních procesů, resp. integračních případů k testovacím příkladům.

Tyto závěry jsou podloženy projektovou dokumentací [NESS, 2006a] (analytickým výstupem, technickým designem řešení a dokumentací testovacích příkladů), která ovšem není veřejná, neboť je majetkem zákazníka a je chráněna obchodním tajemstvím.

17.6.4 Projekt 4 - Budování ODS (2005 - 2006)

Smyslem čtvrtého projektu byla replikace klíčových údajů o zákaznících a jejich produktech do ODS, kde budou k dispozici v prvé fázi přes databázové procedury pro webové aplikace, v druhé fázi budou databázové procedury přístupné přes integrační broker a ve třetí fázi budou poskytovány dalším podnikovým aplikacím. V tomto projektu se řešily tři hlavní integrační úlohy - dávková replikace dat do ODS pro úvodní nalití a případnou dávkovou synchronizaci, online synchronizace změn ze zdrojové databáze do ODS a přístup webových aplikací k ODS. Integrační schéma je na obrázku 72 - Integrační schéma projektu 4 (zdroj: autor).



obrázek 72 - Integrační schéma projektu 4 (zdroj: autor)

V projektu bylo analyzováno a navrženo cca 50 integračních procesů. Analýza, návrh, vývoj i testování probíhal efektivně, neboť díky rozdělení integrační úlohy na integrační případy bylo možno práci distribuovat mezi 7 relativně nezávislých technických týmů, kde každý z nich řešil přidělenou část procesů. Rovněž komunikace se zákazníkem probíhala bez komplikací, neboť veškeré analytické výstupy byly jasně strukturované, přehledné a verifikovatelné.

Ověřené závěry z projektu pro tuto práci:

- Zadání integračního projektu lze definovat formou strukturovaných požadavků v kategoriích, které jsou popsány v metodice.
- Byly ověřeny integrační cesty skrze integrační server, ODS a DSA.
- Referenční model BPI byl ověřen na produktu Tibco.
- Přechod od integračního případu k návrhu jeho řešení integračním postupem a následně detailním návrhem implementace na integrační platformě i na straně aplikací.
- Přechod od integračních procesů, resp. integračních případů k testovacím příkladům.
- Aplikovatelnost základních postupů metodiky na rozsáhlé integrační projekty.

Tyto závěry jsou podloženy projektovou dokumentací [NESS, 2006b] (dodatkem ke smlouvě, analytickým výstupem, technickou specifikací řešení a dokumentací testovacích příkladů), která ovšem není veřejná, neboť je majetkem zákazníka a je chráněna obchodním tajemstvím.

17.7 Terminologický slovník

Termín	Význam	Typ (Nový, Přev-zatý)	Odkaz
Architektura EAI	Integrační rámec, do něhož se v průběhu vývoje informačního systému začleňují jednotlivé aplikace na základě business požadavků a výběru vhodného technologického řešení. Jejím primárním cílem je vytvoření otevřené platformy pro tzv. business integraci, tj. kontinuální zajištění podnikových procesů a jejich vzájemných vazeb s důrazem na stabilitu a adaptabilitu.	P	zdroj [CSSI]
Dceřiná metodika	Metodika, která dědí sdílené prvky mateřské metodiky a rozšiřuje je o specifické vlastnosti pro své vlastní cíle.	N	viz kap. 3
EAI (integrace podnikových aplikací)	Metodický přístup a sada technologických nástrojů pro integraci podnikových procesů.	P	zdroj [CSSI] viz kap. 17.1.3 a kap. 1.1
Externí integrační projekt	Projekt, který řeší integraci aplikací mezi dvěma či více podniky.	P	zdroj [OVUM, 2001]
Integrace	Proces kombinování softwarových komponent, hardwarových komponent, nebo obou, do společného systému.	P	zdroj [LAND, 2004] dle IEEE
Integrační (též přístupový, styčný) bod	Element integračního rozhraní, skrze který aplikace komunikuje se svým okolím, např. konkrétní služba rozhraní.	N	viz kap. 2.1 a 4.5
Integrační cesta	Sada integračních případů, které probíhají mezi dvěma integračními komponentami podnikového informačního systému.	N	viz kap. 11.2
Integrační cesta hlavní	Integrační cestu či kombinace více navazujících cest označíme za hlavní, jestliže v rámci integrační platformy zajišťuje plnění definované integrační úlohy.	N	viz kap. 11.2
Integrační komponenta	Samostatný prvek informačního systému určený primárně pro integraci aplikací.	N	viz kap. 11.2
Integrační koncepce	Tvoří kompaktní (synergickou) kombinaci několika integračních přístupů, které se uplatní současně v daném prostředí, na logické úrovni.	N	viz kap. 10.2
Integrační platforma	V širším pojetí množina integračních komponent implementovaných do PIS. V užším pojetí omezeno na konkrétní integrační komponentu, implicitně BPI.	N	viz kap. 10.2
Integrační postup	Obecný postup nebo i jeho konkrétní implementaci, které jsou primárně určeny pro realizaci integrace aplikací.	N	viz kap. 10.2
Integrační případ	Elementární komunikační operace mezi dvěma integrovanými aplikacemi, nebo aplikací a	N	viz kap. 9

	integrační komponentou. Integrační proces se může skládat z více integračních případů.		
Integrační přístup	Definuje způsob, jak integrovat aplikace za určitých podmínek, na logické úrovni.	N	viz kap. 10.2
Integrační proces	Posloupnost operací, které zajišťují komunikaci s integrovanými aplikacemi, jsou řízeny integrační platformou a mají jeden společný cíl, splněný integračního požadavku. Integrační proces má jasně definovaný začátek (zpravidla příchozí zpráva nebo volání z integrované aplikace, méně často spuštění procesu dle časového plánu) a konec.	N	viz kap. 9
Integrační produkty	Tvoří kompaktní kombinaci několika integračních technologií, které společně tvoří funkční celek na fyzické úrovni.	N	viz kap. 10.2
Integrační projekt	Projekt implementace informačního systému, jehož podstatnou součástí je řešení vybrané integrační úlohy. Integrační úloha řeší komunikaci mezi dvěma a více aplikacemi.	N	viz kap. 2.2
(Integrační) rozhraní	Část aplikace, která je dedikována pro komunikaci s jinými aplikacemi.	N	viz kap. 10.4.1
Integrační standardy	Vybrané technologie, které jsou standardně implementovány v integračních produktech	N	viz kap. 10.2
Integrační strategie	Definuje základní pravidla užití integračních přístupů v rámci jednoho podniku.	N	viz kap. 10.3
Integrační technologie	Je konkrétní implementací integračního přístupu na fyzické úrovni.	N	viz kap. 10.2
Interní integrační projekt	Projekt, který řeší propojení několika aplikací v rámci jednoho podniku.	P	zdroj [OVUM, 2001]
Mateřská metodika	Metodika, jejíž prvky jsou sdíleny pro dceřiné metodiky.	N	viz kap. 3
Metodický rámec	Kolekce metodických vzorů pro různé domény, typy řešení a způsoby řešení spolu s principy a procesy pro vytvoření konkrétní metodiky.	P	[BUCHALCEOVÁ, 2005]
Metodika	Doporučený souhrn etap, přístupů, zásad, postupů, pravidel, dokumentů, řízení, metod, technik a nástrojů pro tvůrce informačních systémů, který pokrývá celý životní cyklus informačních systémů. Definuje kdo, kdy, co a proč má dělat. Poznámka autora: V práci důsledně používám termín metodika, ačkoli v informačních zdrojích se můžeme setkat často i s termínem metodologie. Tyto termíny jsou v oblasti projektů informačních systémů ekvivalentní.	P	[CHLAPEK, 1997], [VOŘÍŠEK]
Middleware	Komponenta, která zajišťuje transparentnost umístění distribuovaných prostředků, tedy že lze pracovat s distribuovanými zdroji jako kdyby byly lokální.	P	zdroj [DOHNAL, 1997]
Model komplexní integrované architektury	Model architektury EAI s důrazem na integrační platformu.	N	viz kap. 11

PIS			
Obchodní (business) pravidlo	Výrok, který definuje nebo omezuje některý aspekt podnikání. Je určen pro uplatnění podnikové struktury nebo ke kontrole nebo vlivu na chování podnikání. Pravidlo je reprezentováno ve formě IF podmínka THEN akce1 ELSE akce2.	P	zdroj [BAJEC, 2005b]
Podniková aplikace	Aplikační software, který vykonává podnikové funkce, jako např. účetnictví, plánování produkce, atd.	P	zdroj [Wikipedia]
Podniková integrace	Cílem podnikové integrace je zajistit včasnou a přesnou výměnu konzistentních informací mezi podnikovými funkcemi pro podporu strategických a taktických podnikových cílů tak, aby se jevila jako nepřerušená.	P	zdroj [SMITH, 2002]
Podnikový (business) proces	Řízená posloupnost činností, spojená definovaným účelem, s cílem vyprodukovat definovaný výstup (produkt, službu). Řízení podnikového procesu je založeno na poznání objektivních zákonitostí a souvislostí typových dějů v rámci oboru činnosti podniku a jejich formalizaci v podobě základního předpisu procesu a pravidel pro jeho řízení.	P	zdroj [CSSI]
(Podnikový) informační systém	Systém, jehož prvky jsou informační a komunikační technologie, data a lidé. Cílem informačního systému je efektivní podpora informačních a rozhodovacích procesů na všech úrovních řízení organizace (podniku).	P	zdroj [CSSI], [VOŘÍŠEK]
Požadavek	Požadovaný rys, vlastnost nebo chování systému.	P	zdroj [FRANK, 2002]
Projekt	1) Dočasná jednorázová práce, která splňuje tato kritéria: o má definovaný začátek a konec, o má časový plán, nákladové a kvalitativní omezení, o je unikátním úsilím a obsahuje rizika, o má jasně daný rozsah, kterého je třeba dosáhnout. 2) Prostorově a časově ohraničený soubor technologicky a organizačně souvisejících činností, jehož účelem je dosažení stanoveného cíle při zadaném čase, zdrojích nákladech a kvalitě.	P	zdroj 1 [CHARVAT, 2003] zdroj 2 [FIALA, 2004]
Projektová dimenze	Řešitelský pohled na projekt. Úlohou řešitelských pohledů na IS/IT je transformovat uživatelské pohledy (požadavky) nejprve do logického návrhu IS/IT, který je nezávislý na použitém technicko-programovém prostředí, a ten pak do fyzického návrhu, který bere ohled na použité technicko-programové prostředí.	P	zdroj [VOŘÍŠEK, 1997]
Projektová fáze	Ucelená část životního cyklu projektu. Každá fáze je charakterizována svými cíli, vstupy,	P	zdroj [VOŘÍŠEK, 1997]

	výstupy a postupem řešení.		
Řízení projektu	Vědomá činnost, směřující k dosažení předem určených cílů projektu, a to v dané kvalitě, v daném čase, v rámci daného rozpočtu a s danými zdroji s tím, že veškeré uvedené parametry se mohou v průběhu projektu měnit podle daných pravidel.	P	zdroj [CSSI]
Systémová integrace	Proces, jehož cílem je vytvoření a další rozvoj komplexního integrovaného informačního systému organizace. Tohoto cíle se dosahuje integrací technologických komponent (technologická integrace), integrací podnikových procesů s funkcemi IS, integrací IS podniku s IS podnikových partnerů, integrací vizí členů vrcholového managementu týkajících se role IS v organizaci a metodickou integrací (vzájemným sladěním metodik využívaných při vývoji a provozu IS).	P	zdroj [CSSI]
Systémový integrátor	Firma, která byla zákazníkem na základě smlouvy pověřena komplexním řešením IS zákazníka. Zodpovídá za kompletní a kvalitní řešení integrovaného IS. K zajištění dodávek může systémový integrátor uzavírat smlouvy s jinými dodavateli.	P	zdroj [CSSI]
Třída produktů	Skupina produktů, které mají obdobnou funkcionalitu a zpravidla jsou v přehledech trhu uváděny společně.	N	viz kap. 17.1.1

17.8 Slovník zkratek

Zkratka	Plný název
A2A	Application to Application
ADO	ActiveX Data Objects
ADSI	Active Directory Services Interface
AIR	Application Integration Relationship
ANSI	American National Standard Institute
API	Application Programming Interface
AREQ	Availability requirements
ARIS	Architecture of Integrated Information Systems
AS	Aplikační server
ASP	podle kontextu 1) Active Server Pages 2) Application Service Provider
ASW	Aplikační software
B2B	Business to Business
B2C	Business to Customer
B2E	Business to Employee
BAM	Business Activity Monitoring
BI	Business Inteligence
BML	Business Model Language
BPEL	Business Process Execution Language
BPI	Business Process Integration server
BPM	Business Process Management
BPMN	Business Process Modeling Notation
BPML	Business Process Modeling Language
BPR	Business Process Reengenering
BRE	Business Rules Engine
CAM	EAI Common Application Metamodel
CBD	Component-Based Development
CDL	Choreography Description Language
CICS	Customer Information Control System
CIM	Computionally Independent Model
CLI	Call Level Interface
CMM	Capability Maturity Model
CMS	Content Management System
COM+	Common (někdy Component) Object Model
CORBA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
DAN	Detailní analýza a návrh
DB	Databáze
DCA	Distributed Component Architecture
DCE	Distributed Computing Environment
DCOM	Distributed Common (někdy Component) Object Model
DMA	Data Mart
DOK	Dokumentační dimenze
DOT	Distributed Object Technologies
DSA	Data Stage Area
DTC	Distributed Transaction Coordinator
DWH	Data Warehouse

EAI	Enterprise Application Integration
EAIF	EAI Framework
EAM	Enterprise Architecture Model
ECA	Enterprise Collaboration Architecture
EDA	Event Driven Architecture
EDI	Electronic Data Interchange
EDIFACT	Electronic Data Interchange for Administration, Commerce and Transport
EDOC	Enterprise Distributed Object Computing
EE	Execution Engine
EI	Enterprise Integration
EII	Enterprise Information Integration
EIA	Enterprise Integration Architecture
EIP	Enterprise Information Portal
EJB	Enterprise Java Beans
EKO	Ekonomická, finanční dimenze
ENS	Enterprise Nervous System
ERP	Enterprise Resource Planning
ESA	Enterprise Service Architecture
ESB	Enterprise Service Bus
ESOA	Extended Service Oriented Architecture
ETL	Extract Transform Load
FCM	Flow Composition Model
FREQ	Functional requirements
FTP	File Transfer Protocol
GAN	Globální analýza a návrh
GST	Globální strategie
GUI	Graphical User Interface
HA	High Availability
HL7	Health Level 7
HLE	High Level Estimation
HTML	Hypertext Modeling Language
HTTP	Hypertext Transfer Protocol
HW	Hardwarová dimenze
IDE	Integrated Development Environment
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IM, IMP	Implementace
IMSL	International Mathematical and Statistical Library
INF, DAT	Informační, datová dimenze
INT	Typ vývoje - integrace řešení
IP	Internet Protocol
IS	Information system
ISDL	Interaction Systems Design Language
ISE	Integrated Service Enviroment
ISO	International Standardization Organization
IST	Informační strategie
IT	Informační technologie
IVR	Interactive Voice Response
J2CA	J2EE Connector Architecture
J2EE	Java 2 Enteprise Edition
JAD	Joint Application Development

JDBC	Java Database Connectivity
JDO	Java Data Objects
JMS	Java Messaging Service
JNDI	Java Naming and Directory Services
JSP	Java Server Pages
JTS	Java Transaction Services
JVM	Java Virtual Machine
KPI	Key Performance Indicator
LCTM	Loosely Coupled Transaction Manager
LDAP	Lightweight Directory Access Protocol
MB	Message broker
MDA	Model Driven Architecture
MeFIS	Methodical Framework for Information Systems Development
MDIS	Methodology of Multidimensional Development of Information System
MET	Metodická dimenze
MMDIS	Multidimensional Management and Development of Information System
MNG	Manažerská dimenze
MOM	Message Oriented Middleware
MREQ	Monitoring requirements
MSMQ	Microsoft Message Queue
NDS	NetWare Directory Services
NEW	Typ vývoje - vývoj nového řešení
OCR	Optical Character Recognition
ODBC	Open Database Connectivity
ODS	Operational Data Store
OGSA	Open Grid Services Architecture
OGSI	Open Grid Services Infrastructure
OIMMDIS	Open Information Model a Meta Data Interchange Specification
OLAP	Online Analytical Processing
OLE	Object Linking and Embedding
ORB	Object Request Broker
OREQ	Others requirements
ORG	Organizační, legislativní dimenze
OS	Operační systém
PHP	PHP: Hypertext Preprocessor
PIM	Platform Independent Model
PIP	Packaged integrating processes
PIS	Podnikový informační systém
PrIS	Programmatic Integration Server
PL/SQL	Procedural Language / Structured Query Language
POC	Proof of Concept
PRA	Pracovní, sociální, etická dimenze
PREQ	Performance requirements
PRO, FUN	Procesní, funkční dimenze
PSM	Platform Specific Model
PU, PUR	Provoz a údržba
QA	Quality Assurance
RAD	Rapid Application Development
RDMS	Relation Database Management System
RFB	Request For Bid
RFD	Request For Demonstration

RFI	Request For Information
RFP	Request For Proposal
RFQ	Request For Quotation
RM-ODP	Reference Model of Open Distributed Processing
RM OSI	Reference Model of Open System Interconnection
RMI	Remote Method Invocation
ROI	Return of Investment
RPC	Remote Procedure Call
SAM	System Architecture Model
SAML	Security Assertions Markup Language
SB	Service Broker
SEC	Bezpečnostní dimenze
SGB	Service Grid Bus
SID	Shared Information/Data Model
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SODA	Service Oriented Development Architecture
SOE	Service Oriented Enterprise
SOI	Service Oriented Integration
SQL	Simple Query Language
SSL	Secure Socket Layer
SSO	Single Sign On
SW	Softwarová dimenze
SWIFT	Society for Worldwide Interbank Financial Telecommunication
TBI	Total Business Integration
TIR	Technical Integration Relationship
TOGAF	The Open Group Architecture Framework
TP	Transaction Processing Monitor
TRPC	Transactional Remote Procedure Call
TYP	Typ vývoje - implementace typového řešení
UAN	Universal Application Network
UDDI	Universal Description, Discovery and Integration Service
UML	Unified Modeling Language
UPG	Typ vývoje - rozvoj a rozšíření řešení
URL	Uniform Resource Locator
US, UST	Úvodní studie
VIP	Very Important Person
VY	Vyřazení
WAP	Wireless Application Protocol
WB	Web Browser
WS	Web Service
WSCI	Web Services Choreography Interface
WSDL	Web Service Definition Language
WSPP	Web Service Producer Platforms
WSRP	Web Services for Remote Portals
WSUI	Web Services User Interface
XAML	Transaction Authority Markup Language
XML	Extensible Markup Language
XP	Extreme Programming
ZA, ZAV	Zavádění

17.9 Použité zdroje

1. ACHARYA, R. 2003. *EAI: A Business Perspective*. eAI Journal [online], 2003, April. Dostupný na WWW: <http://www.eaijournal.com>.
2. ALAM, H. 2001.*Understanding Integration Strategies*. eAI Journal [online], 2001, November. Dostupný na WWW: <http://www.eaijournal.com>.
3. ALBIN, S. T. 2003. *The Art of Software Architecture: Design Methods and Techniques*. 1st edition. Wiley, 2003. 312 s. ISBN 0471228869.
4. ALTMAN, R. 2003a. *Forward Thinking - Integrating Your Integration Tools?.* Business Integration Journal [online], 2003a, August. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
5. ALTMAN, R. 2003b. *Forward Thinking - What's the ROI for EAI.* Business Integration Journal [online], 2003, September. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
6. ARLOW, J., NEUSTADT, I. 2003. *Enterprise Patterns and MDA: Building Better Software with Archetype Patterns and UML*. 1st edition. Addison Wesley, 2003. 528 s. ISBN 0-321-11230-X.
7. ARPINAR, I. B., et al. 2005. *Ontology-Driven Web Services Composition Platform*. Information Systems and E-Business Management [online]. Springer, 2005. DOI 10.1007/s10257-005-0055-9.
8. *ARRIBA: GBOU project proposal* [online]. Dostupný na WWW: <http://arriba.vub.ac.be/project.htm>.
9. ATKINSON, M. P., et al. 2003. *Grid Database Access and Integration: Requirements and Functionalities* [online]. Global Grid Forum, 2003. Dostupný na WWW: <http://www.gridforum.org>.
10. ATTACHMATE. *Approaches to EAI Involving Legacy Host Applications: The Five R's* [online]. Dostupný na WWW: <http://www.attachmate.com/article/0,1012,3163_1_3858,00.html>.
11. BABU, M. 2003. *Four Questions Business Managers Should Ask About EAI*. eAI Journal [online], 2003, April. Dostupný na WWW: <http://www.eaijournal.com>.
12. BAJEC, M., KRISPER, M. 2005a. *A methodology and tool support for managing business rules in organisations (University of Ljubljana, Faculty of Computer & Information Science)*. Information Systems, 2005, September, vol. 30, no. 6, s. 423-443. ISSN 0306-4379.
13. BAJEC, M., KRISPER, M. 2005b. *Issues and Challenges in Business Rule-based Information System Development (University of Ljubljana, Faculty of Computer & Information Science)*. In European Conference on Information Systems, Regensburg (Germany), 2005. Regensburg (Germany). Dostupný na WWW: <http://aisel.isworld.org/article_by_authors.asp?Author_ID=6221>
14. BARBASH, B. R. *Fiorano ESB - A strong enterprise service bus solution*. Web Services Journal [online], Dostupný na WWW: <http://webservices.sys-con.com/read/39931.htm>.
15. BARLAS, D. 2003. *Why EAI Fails.* Line56 [online], 2003, http://www.line56.com/articles/default.asp?ArticleID=5232.
16. BARRY, D. K. 2003. *Web Services and Service-Oriented Architecture: The Savvy Manager's Guide*. 1st edition. Morgan Kaufmann, 2003. 245 s. ISBN 1558609067.
17. BASS, Ch., LEE, J. M. 2002. *Building a Business Case for EAI*. eAI Journal [online], 2002, January. Dostupný na WWW: <http://www.eaijournal.com>.
18. BASS, L., CLEMENTS, P., KAZMAN, R. 2003. *Software Architecture in Practice*. 2nd edition. Addison Wesley, 2003. 560 s. ISBN 0-321-15495-9.
19. *BEA AquaLogic Service Bus™ Concepts and Architecture 2.0*. BEA Systems, 2005. Dostupný na WWW: <http://e-docs.bea.com>.

20. *BEA Liquid Data for Weblogic.* BEA Systems. Dostupný na WWW: <http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/liquid_data>.
21. *BEA WebLogic Integration 8.1 - Product Information*. BEA Systems. Dostupný na WWW: <http://e-docs.bea.com/wli/docs81/index.html>.
22. BÉBR, R., DOUCEK, P. 2005. *Informační systémy pro podporu manažerské práce*. 1.vydání. Proffesional Publishing, 2005. 224 s. ISBN 0-87-389-643-2
23. BECK, K. 2002. *Test-Driven Development By Example*. 1st edition. Addison Wesley, 2002. 240 s. ISBN 0-321-14653-0.
24. BECK, K., FOWLER, M. 2000. *Planning Extreme Programming*. 1st edition. Addison Wesley, 2000. 198 s. ISBN 0-201-71091-9.
25. BHATIA, K. 2005. *Peer-To-Peer Requirements On The Open Grid Services Architecture Framework* [online]. Global Grid Forum, 2005. Dostupný na WWW: <http://www.gridforum.org>.
26. BIEBERSTEIN, N., et al. 2005. *Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap*. 1st edition. IBM Press, 2005. 272 s. ISBN 0-13-187002-5.
27. BISHOP, T. A., KARNE, R. K. 2003. *A Survey of Middleware (Towson University)*. In 18th International Conference on Computers and Their Applications, Hawaii, 2003. Dostupný na WWW: <http://triton.towson.edu/~karne/research/middlew/>.
28. BLEVINS, T. 2004. *TBI and TOGAF Reconciliation*. Global EAI Summit, 2004. Dostupný na WWW: <http://www.opengroup.org/conference-live/uploads/40/4511/EAIIC-TOG-Blevins.pdf>.
29. BOND, A. 2001. *Middleware and Enterprise Architectures* [online]. 2001, Dostupný na WWW: <http://www.dstc.edu.au/Research/Projects/Integrator/>.
30. *BPI - Integrace podnikových procesů*. IT System, 2002, č.10. ISSN 1212-4567.
31. BROWN, T. M. 2002. *Web Services - EAI Killer ?*. eAI Journal [online], 2002, April. Dostupný na WWW: <http://www.eaijournal.com>.
32. BUCHALCEOVÁ, A. 2003. *Model Driven Architecture jako nový přístup k vývoji i integraci aplikací*. In System Integration 2003. Praha : VŠE KIT, 2003, s. 469-475. ISBN 80-245-0522-3.
33. BUCHALCEOVÁ, A. 2005. *Metodiky vývoje a údržby informačních systémů: Kategorizace, agilní metodiky, vzory pro návrh metodik*. 1. vydání. Praha: Grada, 2005. 164 s. ISBN 80-247-1075-7.
34. BURANSKÝ, I. 2005. [online]. *XML a webové služby*. 1. vydání. Microsoft, 2005. Dostupný na WWW: <http://msdn.microsoft.cz/ docs/BrozuraXML.zip>.
35. BUSSLER, Ch. 2002a. *B2B Integration Technology Architecture (Oracle Corporation)*. In 4th IEEE Int'l Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS 2002) (IEEE), 2002. Dostupný na WWW: <http://doi.ieeecomputersociety.org/10.1109/WECWIS.2002.1021252>.
36. BUSSLER, Ch. 2002b. *P2P in B2BI (Oracle Corporation)*. In 35th Annual Hawaii International Conference on System Sciences (HICSS-02) - Volume 9 (IEEE), Hawaii, 2002. Dostupný na WWW: <http://csdl.computer.org/comp/proceedings/ hicss/2002/1435/09/14350302.pdf>.
37. BUSSLER, Ch. 2003. *B2B Integration: Concepts and Architecture*. 1st edition. Springer, 2003. 440 s. ISBN 3-540-43487-9.
38. CAREY, M. 2005. *Data Services: This Is Your Data on SOA.* Business Integration Journal [online], 2005, November/ December. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
39. CARL, A. 2003. *10 Keys to Successful EAI*. eAI Journal [online], 2003, March. Dostupný na WWW: <http://www.eaijournal.com>.

40. CAVELTI, S. 2004. *Joint Business-Driven and Data-Driven Enterprise Application Integration in Industry*. Diplomová práce, Institut für Informatik der Universität Zürich, 2004. Matrikelnummer 98 918 691.
41. CERAMI, E. 2002. *Web Services Essentials, Distributed Applications with XML-RPC, SOAP, UDDI & WSDL*. 1st edition. O'Reilly, 2002. 304 s. ISBN 0-596-00224-6.
42. CIMETIERE, J. Ch. 2003. *Adoption of Web Services & Technology Choices*. TechMetrix Research, February 2003, 70 s.
43. CLAUNCH, C. 2003. *Top 10 Strategic Technologies*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
44. CLAY, R. *Survey: Application Integration Survey*. Integration Consortium. 30 s. Dostupný na WWW: <http://www.integrationconsortium.org/ docs/member%20docs/SurveyResults.pdf>.
45. COCKBURN, A. 2001. *Agile Software Development.* 1st edition. Addison Wesley, 2001. 256 s. ISBN 0201699699.
46. COHEN, F. 2004. *Java Testing, Design and Automation: A Guide to Maximizing Performance*. 1st edition. Prentice Hall, 2004. 434 s. ISBN 0131421891.
47. CORBA. *CORBA vendors* [online]. CORBA. Dostupný na WWW: <http://www.corba.org/vendors/>.
48. CRAGGS, S. 2005. *Navigating the Integration in Minefield.* Business Integration Journal [online], 2005, May. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
49. CRISPIN, L., HOUSE, T. 2002. *Testing Extreme Programming*. 1st edition. Addison Wesley, 2002. 336 s. ISBN 0-321-11355-1.
50. CSC. 2001. *Business Integration Practice Guide* [interní materiál]. CSC Computer Sciences, 2001.
51. CSC. 2003. *Dokumentace k projektu eWatch* [interní materiál]. CSC Computer Sciences, 2003.
52. CSSI. *Terminologický slovník České společnosti pro systémovou integraci* [online]. Dostupný na WWW: <http://www.cssi.cz/all_terminologie.asp>.
53. CZADEK, M., BENEŠ, J. 2004. *EAI - Principy a faktory ovlivňující výběr řešení integrace aplikací*. In System Integration 2004. Praha : VŠE KIT, 2004, s. 219-227. ISBN 80-245-0701-3.
54. DEEB, J., NARTH, P. 2005. *A Match Made in Heaven? Combining EAI and ETL to Achieve the Single Source of Truth.* Business Integration Journal [online], 2005, September. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
55. DEVARAJ, S. 2004. *Enterprise Integration Styles 101*. SETLabs Briefings [online], Infosys, 2004, October/ December, vol. 2, no. 4 (Enterprise Architecture & Business Competitiveness), s. 59-67, Dostupný na WWW: <http://www.infosys.com/technology/toc.asp>.
56. DOHNAL, J., POUR, J. 1997. *Architektury informačních systémů v průmyslových a obchodních podnicích*. 1. vydání. Praha: Ekopress, 1997. 301 s. ISBN 8086119025.
57. DOLGICER, M., BAYER, G. 2004. *Message Brokers vs. Application Servers for Integration.* Business Integration Journal [online], 2004, March. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
58. DONNELLY, T., DENMAN, M. 2004. *Oracle Application Server 10g*. Oracle, 2004. Dostupný na WWW: <http://www.gcoug.org/presentations/>.
59. DUNCAN, W. R. 1996. *A Guide to the Project Management Body of Knowledge*. 1st edition. Project Management Institute, 1996. 182 s. ISBN 1-880410-12-5.
60. DUSTIN, E. 2002. *Effective Software Testing: 50 Specific Ways To Improve Your Testing*. 1st edition. Addison Wesley, 2002. 203 s. ISBN 0-201-79429-2.
61. ELLIS, D. H. a. *Benefits of Using a Central EAI Hub to Support Web Services* [online]. State of Maine, Bureau of Information Services. Dostupný na WWW: < http://www.maine.gov/ ec/eaireview/eaibenefits.htm>.

62. ELLIS, D. H. b. *Enterprise Application Integration for Maine - A review* [online]. State of Maine, Bureau of Information Services. Dostupný na WWW: <http://www.maine.gov/ec/eaireview/fullintreview.htm>.
63. ENDLICH, U. 2004. *An Evaluation of Enterprise Application Integration Solutions and the Role of Web Services*. Diplomová práce, Institut für Verteilte Systeme der Technischen Universität, Wien, 2004.
64. DIAMONDCLUSTER. 2001. *Enterprise Application Integration*. Center for Technology Innovation / DiamondCluster, 2001.
65. EAI INDUSTRY CONSORTIUM. 2004. *Total Business Integration Methodology v1.3*. EAI Industry Consortium, 2004. Dostupný na WWW: <http://www.xml.gov/presentations/ic/tbi.htm>.
66. ERASALA, N., YEN, D. C., RAJKUMAR, T. M. 2003. *Enterprise Application Integration in the electronic commerce world (Miami University).* Computer Standards & Interfaces, 2003, no. 25, s. 69–82. ISSN 0920-5489.
67. ERL, T. 2005. *Service-Oriented Architecture: Concepts, Technology, and Design*. 1st edition. Prentice Hall, 2005. 792 s. ISBN 0-13-185858-0.
68. FARGES, N. 2003. *Enterprise Service Bus (ESB): Lasting concept or latest buzzword?*. TechMetrix Research, April 2003.
69. FENN, J. 2003. *2003 Technology Radar Screen: Emerging Trends and Technologies Scenario*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
70. FEUERLICHT, G. 2001. *Business Middleware Standards and Architectures*. In System Integration 2001. Praha : VŠE KIT, 2001, s. 265-271. ISBN 80-245-0169-4.
71. FIALA, P. 2004. *Projektové řízení: Modely, metody, analýzy*. 1.vydání. Praha: Grada, 2004. ISBN 80-86419-24-X.
72. FOSTER, I., et al. 2002. *The Physiology of the Grid An Open Grid Services Architecture for Distributed Systems Integration* [online]. Globus, 2002. Dostupný na WWW: <http://www.globus.org/research/papers/ogsa.pdf>.
73. FOSTER, I., et al. 2005a. *The Open Grid Services Architecture, v1.0* [online]. Global Grid Forum, 2005. Dostupný na WWW: <http://www.gridforum.org>.
74. FOSTER, I., TUECKO, S. 2005b. *Describing the Elephant: The Different Faces of IT as Service*. ACM Queue [online], 2005, July/ August, vol. 3, no. 6. Dostupný na WWW: <http://www.acmqueue.com>. ISSN 1542-7730.
75. FOURNIER-MOREL, X. 2004. *Enterprise Information Integration: Hype or reality?*. TechMetrix Research, January 2004.
76. FOWLER, M. *Continuous Integration* [online]. Dostupný na WWW: <http://martinfowler.com/articles/ continuousIntegration.html>.
77. FOWLER, M., et al. 2002. *Patterns of Enterprise Application Architecture*. 1st edition. Addison Wesley, 2002. 560 s. ISBN 0-321-12742-0.
78. FOX, G., et al. 2003. *Overview of Grid Computing Environments* [online]. Global Grid Forum, 2003. Dostupný na WWW: <http://www.gridforum.org>.
79. FRANK, B., MARRIOT, P., WAZUSEN, Ch. 2002. *The Software Quality Engineer Primer*. 3rd edition. Quality Council of Indiana, 2002. B0006QSVWW.
80. FRENTZEN, J. 2003. [online]. *Database Buying Guide.* 5th edition. Aberdeen Group, 2003. 177 s. Dostupný na WWW: <http://www.aberdeen.com>.
81. FRIEDMAN, T. 2003. *ETL Magic Quadrant Update: Market Pressure Increases*. Gartner, January 2003.
82. GANDHI, A., SALA, L. 2005. *AquaLogic Service Registry - An Enterprise-Class Solution for SOA Lifecycle Management*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.
83. GARTNER. 2003. *Application Platform Suites - An Architectural Cost Analysis*. Gartner & BEA Systems, November 2003.
84. GAWLICK, D. *The Database as The Application Platform* [online]. Dostupný na WWW: <http://www.research.microsoft.com/~gray/hpts99/papers/Gawlick.htm>.

85. GILPIN, M., et al. 2004. *What Is An Enterprise Service Bus?*. Forrester Research, 2004. 6 s.
86. GOLD, R., HAMMELL, T., SNYDER, T. 2004. *Test-Driven Development: A J2EE Example*. 1st edition. Apress, 2004. 296 s. ISBN 1-59059-327-8.
87. GOLDSTONE. *Enterprise Application Integration - An Overview*. GOLDSTONE Technologies Limited. Dostupný na WWW: <www.goldstonetech.com/downloads/EAI%20Overview.pdf >.
88. GROENENDAAL, W. 2002. *The Integration Project Lifecycle*. IPA Spring Days on Middleware, April 2002. Dostupný na WWW: <http://www.win.tue.nl/ipa/archive/springdays2002/>.
89. HALL, D., BAILEY, W., GEORGE, J. 2005. *A Service-Oriented Enterprise Strategy for Enterprise Integration: Part II - The Structural Elements & Disciplines.* Business Integration Journal [online], 2005, September. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
90. HALLE, von, B. 2002. *Business Rules Applied, Building Better Systems Using the Business Rules Approach*. 1st edition. New York: Wiley, 2002. 495 s. ISBN 0-471-41293-7.
91. HASS, A. M. J. 2002. *Configuration Management Principles And Practice*. 1st edition. Addison-Wesley, 2002. 432 s. ISBN 0-321-11766-2.
92. HAUBRICH, M. 2005. *Pre-Evaluating Application Integration Projects*. Diplomová práce, Vrije Universiteit Amsterdam, Faculty of Sciences, Dept. of Computer Science, 2005.
93. HELLER, T. 2003. *Logical Portal Architecture*. CommITment Consulting Framework, 2003. Dostupný na WWW: <http://www.commitment.no>.
94. HERRERA, J. 2003. *Organizational Readiness for EAI*. eAI Journal [online], 2003, March. Dostupný na WWW: <http://www.eaijournal.com>.
95. HOFFMANN, K. 2005. *BEA Weblogic Integration Patterns and Best Practices for SOA*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.
96. HOHMANN, L. 2003. *Beyond Software Architecture: Creating and Sustaining Winning Solutions*. 1st edition. Addison Wesley, 2003. 352 s. ISBN 0-201-77594-8.
97. HOHPE, G. 2002a. *Stairway to Heaven* [online]. Software Development, 2002, Dostupný na WWW: <http://www.hohpe.com/Gregor/Work/>.
98. HOHPE, G. 2003a. *Enterprise Integration Patterns - Asynchronous Messaging Architectures in Practice*. ThoughtWorks, 2003.
99. HOHPE, G. *Enterprise Integration Patterns* [online]. Dostupný na WWW: <http://www.eaipatterns.com/>.
100. HOHPE, G., ISTVANICK, W. 2002b. *Test-Driven Development in Enterprise Integration Projects*. ThoughtWorks, 2002.
101. HOHPE, G., THAM, H. S. 2004. *Enterprise Integration Patterns with BizTalk Server 2004*. ThoughtWorks, 2004.
102. HOHPE, G., WOOLF, B. 2003b. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. 1st edition. Addison Wesley, 2003. 736 s. ISBN 0-321-20068-3.
103. HORCH, J. W. 2003. *Practical Guide to Software Quality Management.* 2nd edition. Artech House, 2003. 286 s. ISBN1580535275.
104. HOSTBRIDGE. 2003. *Integrating Legacy Applications as Web Services*. Hostbridge Technologies, May 2003. Dostupný na WWW: <www.hostbridge.com/white_papers/web_ services_general/webservicesGeneralWhitePaper.pdf>.
105. HUTCHESON, M. L. 2003. *Software Testing Fundamentals: Methods and Metrics*. 1st edition. Wiley, 2003. 408 s. ISBN 047143020X.

106. CHAKRAVARTHY, S. 2004. *ESB Best Practices Best Practices.* Fiorano Software, 2004.
107. CHAPPELL, D. 2004. *Enterprise Service Bus*. 1st edition. O'Reilly, 2004. 274 s. ISBN 0-596-00675-6.
108. CHARVAT, J. 2003. *Project Management Methodologies: Selecting, Implementing, and Supporting Methodologies and Processes for Projects*. 1st edition. Wiley, 2003. 264 s. ISBN 0-471-22178-3.
109. CHIKARMANE, S. 2005. *BEA Integration Products Roadmap*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.
110. CHLAPEK, D. 2005. *Základní objekty metodického rámce pro řízení a koordinaci projektů IS/ICT*. In System Integration 2005. Praha : VŠE KIT, 2005, s. 272-278. ISBN 80-245-0895-8.
111. CHLAPEK, D., ŘEPA, 1997. V. *Materiály ke strukturované analýze*. 1. vydání. Praha: Vysoká škola ekonomická, 1997. 138 s. ISBN 80-7079-260-4.
112. IBARRA, F. *The Enterprise Service Bus: Building Enterprise SOA*. BEA Systems. Dostupný na WWW: <http://dev2dev.bea.com/technologies/soa/articles/soa_ibarra.jsp>.
113. IBM. *Application Integration General Guidelines* [online]. IBM. Dostupný na WWW: <http://www-106.ibm.com/developerworks/patterns/application/>.
114. ILKAEV, D., MEENAN, D. *Analysis of the Specifics for a Business Rules Engine Based Projects* [online]. Dostupný na WWW: <http://http://java.ittoolbox.com/white-papers/Analysis-of-the-Specifics-for-a-Business-Rules-Engine-Based-Projects-3486>.
115. IMHOFF, C., GREEN, J. 2004. *The Three E's: EAI, ETL and EII - Which to Use, and When?* [webinar]. Inteligent Solutions, 2004. Dostupný na WWW: <http://www.ebizq.net/webinars/5551.html/?comp>.
116. *Integration Consortium* [online]. Dostupný na WWW: <http://www.integrationconsortium.org>.
117. IPEDO. 2005. *Guide to Enterprise Information Integration (EII) - An Overview of EII Technology and How to Use It*. Ipedo, 2005. Dostupný na WWW: <http://www.ipedo.com/eii_guide>.
118. JACKSON T., WHEAT, S. 2003. *OpenEAI Implementation Strategies 1.0* [online]. OpenEAI Software Foundation, 2003. Dostupný na WWW: <http://www.openeai.org>.
119. JACKSON, T., MAJUMDAR, R., WHEAT, S. 2005. *OpenEAI Methodology 1.0* [online]. OpenEAI Software Foundation, 2005. Dostupný na WWW: <http://www.openeai.org>.
120. JAISWAL, R. *Security Concerns in EAI*. WiPro Technologies. Dostupný na WWW: <http://www.wipro.com/insights/securityeai.htm>.
121. JALOTE, P. 2002. *Software Project Management in Practice*. 1st edition. Addison Wesley, 2002. 288s. ISBN 0-201-73721-3.
122. JANDOŠ, J. 2004. *K integraci aplikací*. In System Integration 2004. Praha: VŠE KIT, 2004, s. 115–126. ISBN 80-245-0701-3.
123. JANDOŠ, J. 2005. *Integration - supporting business goals*. In System Integration 2005. Praha : VŠE KIT, 2005, s. 279-283. ISBN 80-245-0895-8.
124. JANSSEN, M., CRESSWELL, A. 2005. *Enterprise Architecture Integration in E-government (Delft University of Technology, University at Albany-SUNY)*. In 38th Annual Hawaii International Conference on System Sciences (HICSS-05) (IEEE), Hawaii, 2005. Dostupný na WWW: <http://csdl2.computer.org/comp/proceedings/hicss/2005/2268/05/22680118b.pdf>.
125. JEAN-CHARLES, A., JOSHI, S. D. 2001. *Architectural Choices for OSS Integration*. eAI Journal [online], 2001, September. Dostupný na WWW: <http://www.eaijournal.com>.
126. JEFFRIES, R., et al. 2000. *Extreme Programming Installed*. 1st edition. Addison Wesley, 2000. 288 s. ISBN 0201708426.

127. JOHANNESSON, P., WANGLER, B., JAYAWEERA, P. 2000. *Application and Process Integration: Concepts, Issues, and Research Directions (Stockholm University, University of Skövde)*. In Information Systems Engineering Symposium CAiSE, 2000. Dostupný na WWW: <http://www.dsv.su.se/~perjons/fossilpaul2.pdf>.
128. JOHANNESSON, P., PERJONS, E. 2001. *Design principles for process modeling in enterprise application integration*. Information Systems, 2001, no. 26. s. 165-184.
129. JONES, T. 2003. *Enterprise Application Integration Technology Audit*. Butler Group, 2003. 8 s. TA000294EAI.
130. JURIC, M. B., et al. 2001. *Professional J2EE EAI*. 1st edition. Wrox, 2001, 1000 s. ISBN 186100544X.
131. JURIC, M. B. 2002. *EAI and Web Services*. eAI Journal [online], 2002, July. Dostupný na WWW: <http://www.eaijournal.com>.
132. JURÍK, P., LÁVIČKA, J., SOBOTKA, M. 2003. *Integrace podnikových aplikací za pomoci Microsoft BizTalk Serveru 2002*. Microsoft, 2003. 229 s. Dostupný na WWW: <http://www.pcweek.sk/downloads/details.php?id=42>.
133. JUŘEK, M. 2004. *Moderní integrace aplikací*. Microsoft, 2004. 114 s. Dostupný na WWW: <http://www.microsoft.cz/akce/msdn_brozury/PdfDownload.asp?download=download_moderniintegrace>.
134. JUŘEK, M. 2005. *Integrace, interoperabilita a budoucnost* [online prezentace z konference]. Microsoft EMEA, 2005. Dostupný na WWW: <http://www.vyvojar.cz/Default.aspx?tabid=1&newsType=CategoryView&CategoryID=11>.
135. KAN, S. H. 2002. *Metrics and Models in Software Quality Engineering*. 2nd edition. Addison Wesley, 2002. 560 s. ISBN 0-201-72915-6.
136. *Kapow RoboSuite, BEA WebLogic Edition*. BEA Systems. Dostupný na WWW: <http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/third/kapow/>
137. KARAS, J. 2001. *Model e-firmy a analýza bezpečnostních rizik e-podnikání*. Diplomová práce, Vysoká škola ekonomická v Praze, Katedra informačních technologií, Praha, 2001.
138. KARAS, J. 2002. *Využití middleware při integraci s důrazem na bezpečnost*. Systémová integrace, Praha, 2002, ročník 9, číslo 1. ISSN 1210-9479.
139. KARAS, J. 2004. *Kritéria výběru produktů EAI*. In Sborník prací účastníků vědeckého semináře doktorského studia fakulty Informatiky a statistiky VŠE v Praze. Praha: VŠE FIS, 2004, s. 67-78. ISBN 80-245-0706-4.
140. KARAS, J. 2006a. *Integrační testování*. In Sborník prací účastníků vědeckého semináře doktorského studia fakulty Informatiky a statistiky VŠE v Praze. Praha: VŠE FIS, 2006, s. 39-47. ISBN 80-245-1037-5.
141. KARAS, J. 2006b. *Integrační procesy v kontextu podnikových procesů*. In System Integration 2006. Praha: VŠE KIT, 2006, s. 521–529. ISBN 80-245-1050-2.
142. KASHYAP, V., SHETH, A. 2000. *Information Brokering across Heterogeneous Digital Data: A Metadata-based Approach*. 1st edition. Springer, 2000. 248 s. ISBN 0792378830.
143. KAYE, D. 2003. *Loosely Coupled: The Missing Pieces of Web Services*. 1st edition. RDS Press, 2003. 334 s. ISBN 1881378241.
144. KEEN, M. 2004. *Patterns: Implementing an SOA Using an Enterprise Service Bus*. IBM, 2004. 386 s. Dostupný na WWW: <http://ibm.com/redbooks>.
145. KERNOCHAN, W. 2003. [online]. *Enterprise Information Integration - The New Way to Leverage E-information*. 2nd edition. [s.l.]: Aberdeen Group, 2003. 123 s. Dostupný na WWW: <http://www.aberdeen.com>.
146. KHANNA, R. 2004. *Top Challenges in Integration Projects*. WiPro Technologies, 2004. Dostupný na WWW: <http://hosteddocs.ittolbox.com/WT052004.pdf>.

147. KIMBALL, R., ROSS, M. 2002. *The Datawarehouse Toolkit*. 2nd Edition. Wiley, 2002. 416 s. ISBN 0471200247.
148. KING, N. 2004. *Getting it all together* [online]. ComputerUser, 2004, November 2004. Dostupný na WWW: <http://www.computeruser.com/clickit/printout/articles/5546400,1248,6,1,1101,3424377600004.html>.
149. KOVACIC, A., GROZNIK, A., KRISPER, M. 2002. *Business Renovation: From Business Process Modelling to Information System Modelling*. I. J. of Simulation, University of Ljubljana, Faculty of Economics, 2002, vol. 2, no. 2. ISSN 1473-804x.
150. KRAFZIG, D., BANKE, K., SLAMA, D. 2004. *Enterprise SOA: Service-Oriented Architecture Best Practices*. 1st edition. Prentice Hall, 2004. 408 s. ISBN 0131465759.
151. KRAUS, J. 2002. *Enterprise Application Integration (EAI) – Ein Anwenderleitfaden für die IT-Abteilung einer deutschen Großbank*. Diplomová práce, Technische Universität München, Fakultät für Informatik, 2002.
152. KRILL, P. 2004. *BEA holds app server market lead but revenues drop.* InfoWorld, 2004, July. ISSN 0199-6649.
153. KRISHNAN, M. *The EAI Paradigm Shift*. WiPro Technologies. Dostupný na WWW: <http://www.wipro.com/insights/eaiparadigm.htm>.
154. KRUCHTEN, P. 2003. *The Rational Unified Process: An Introduction*. 3rd edition. Addison Wesley, 2003. 336 s. ISBN 0-321-19770-4.
155. KUTÁČ, D., ZUBEK, M. 2004. *Ensemble - integrační produkt nové generace*. In System Integration 2004. Praha : VŠE KIT, 2004, s. 235-243. ISBN 80-245-0701-3.
156. LAM, W., SHANKARARAMAN, V. 2004. *An Enterprise Integration Methodology (IEEE)*, IT Professional, 2004, March/ April, vol. 06, no. 2, s. 40-48. ISSN 1520-9202.
157. LAMONICA, M. a. *IBM plans to debut the first fruits of a long-standing research project later this year, setting the stage for a clash with rivals in the multibillion-dollar database software market*. News.com [online]. Dostupný na WWW: <http://news.com.com/2100-1001-980196.html>
158. LAMONICA, M. b. *Market research gives Oracle top database spot*. News.com [online]. Dostupný na WWW: <http://news.com.com/Market+research+gives+Oracle+top+database+spot/2100-7343_3-5227856.html>.
159. LAND, R., CRNKOVIC, I. 2004. *Existing Approaches to Software Integration – and a Challenge for the Future (Mälardalen University, Department of Computer Science and Engineering)*. In 4th Conference on Software Engineering Research and Practice, Sweden, 2004. Dostupný na WWW: <http://citeseer.ist.psu.edu/719316.html>.
160. LANGER, M., NOVOTNÝ, R. 2004. *Záludnosti EAI*. In System Integration 2004. Praha : VŠE KIT, 2004, s. 583-588. ISBN 80-245-0701-3.
161. LAROIA, A., SAYAVEDRA, L. Jr. 2003. *EAI Business Drivers*. eAI Journal [online], 2003, February. Dostupný na WWW: <http://www.eaijournal.com>.
162. LAWLER, J., et al. 2005. *A Study of Web Services Strategy in the Financial Services Industry (Pace University)*. Information Systems Education Journal [online], 2005, 3/3. Dostupný na WWW: <http://isedj.org/isecon/2004/3443/>. ISSN 1545-679X.
163. LAYTON, C. *Middleware Technology* [online]. Rogers State University. Dostupný na WWW: <http://www.faculty.rsu.edu/~clayton/ns/middleware2.doc>.
164. LEE, R. 2005. *BEA Tuxedo® Roadmap 9.0 and Beyond*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.
165. LEFFINGWELL, D., WIDRIG, D. 2005. *Managing Software Requirements: A Use Case Approach.* 2nd edition. Addison Wesley, 2005. 544 s. ISBN 0-321-12247-X.
166. LEVINE, R. 2002. *Estimating the Scope of Your EAI Project*. eAI Journal [online], 2002, July. Dostupný na WWW: <http://www.eaijournal.com>.
167. LHEUREUX, B. 2003. *Middleware 101: The Cheat Sheet*. Gartner Symposium ITxpo 2003, Gartner, March 2003.

168. LINDSTEDT, D. E. 2005. *What does ETL do that EAI Can't?* [online]. 2005. Dostupný na WWW: <http://www.b-eye-network.com/blogs/linstedt/archives/2005/11/what_does_etl_d.php>.
169. LINEHAN, M. H., FERGUSON, D. F. 2005. *Business Rule Standards - Interoperability and Portability*; W3C Workshop on Rule Languages for Interoperability 2005. Dostupný na WWW: <http://www.w3.org/2004/12/rules-ws/paper/113/>.
170. LINKIN, P. 2004. *The Convergence of Application Development and Integration.* Business Integration Journal [online], 2004, May. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
171. LINTHICUM, D. S. 1999. *Enterprise Application Integration*. 1st edition. Addison Wesley, 1999. 400 s. ISBN 0-201-61583-5.
172. LINTHICUM, D. S. 2002. *The Evolution of Adapters*. eAI Journal [online], 2002, December. Dostupný na WWW: <http://www.eaijournal.com>.
173. LINTHICUM, D. S. 2003. *Next Generation Application Integration*. 1st edition. Addison Wesley, 2003. 512 s. ISBN 0-201-84456-7.
174. LINTHICUM, D. S. 2004a. *The Next Wave - Application Integration and the Supply Chain: Integrating the Notions - Part 1.* Business Integration Journal [online], 2004, March. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
175. LINTHICUM, D. S. 2004b. *The Next Wave - Application Integration and the Supply Chain: Integrating the Notions - Part 2.* Business Integration Journal [online], 2004, April. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
176. LITTLE, T. 2005. *Web Services in BEA Tuxedo 9.0*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.
177. LOSAVIO, F., ORTEGA, D., PÉREZ, M. 2002. *Modeling EAI.* In 12th International Conference of the Chilean Computer Science Society (SCCC'02), Chile, 2002. Dostupný na WWW: <http://doi.ieeecomputersociety.org/10.1109/SCCC.2002.1173194>.
178. LOSAVIO, F., ORTEGA, D., PÉREZ, M. 2003. *Towards a Standard EAI Quality Terminology*. In 23rd International Conference of the Chilean Computer Science Society (SCCC'03) (IEEE), Chile, 2003. Dostupný na WWW: <doi.ieeecomputersociety.org/10.1109/SCCC.2003.1245452>.
179. LOSAVIO, F., ORTEGA, D., PÉREZ, M. 2005. *Comparison of EAI Frameworks*. Journal of Object Technology [online], 2005, May/ June, vol. 4, no. 4, s. 93-114. Dostupný na WWW: <http://www.jot.fm/issues/issue_2005_05/article1>. ISSN 1660-1769.
180. LUBLINSKY, B. 2001. *Achieving the Ultimate EAI Implementation. Part 2: Message-Level Integration*. Business Integration Journal [online], 2001, January. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
181. LUBLINSKY, B., FARRELL, M. Jr. 2002. *Top 10 Reasons Why EAI Fails*. eAI Journal [online], 2002, December. Dostupný na WWW: <http://www.eaijournal.com>.
182. LUTZ, J. C. 2000. *EAI Architecture Patterns*. eAI Journal [online], 2000, March. Dostupný na WWW: <http://www.eaijournal.com>.
183. MACVITTIE, D. 2003. *Smooth Integrators*. Network Computing [online], 2003, 8.7., s. 37-46. Dostupný na WWW: <http://www.networkcomputing.com/gswelcome/showArticle.jhtml?articleID=15000693&pgno=1>. ISSN 1046-4468.
184. MALVEAU, R., MOWBRAY, T. J. 2000. *Software Architect Bootcamp*. 1st edition. Prentice Hall, 2000. 352 s. ISBN 0-13-027407-0.
185. MARCUS, R. 2002. *Great Global Grid: Emerging Technology Strategies*. 1st edition. Trafford Publishing, 2002. 352 s. ISBN 1-55369-884-3.
186. MATĚJŮ, D. 2005. *Automatizace podnikových procesů založená na standardech*. In System Integration 2005. Praha : VŠE KIT, 2005, s. 68-71. ISBN 80-245-0895-8.

187. MAVERICK, G. 2003. *EAI Project Management.* Business Integration Journal [online], 2003, November. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
188. MCCONNELL, S. 2003. *Professional Software Development: Shorter Schedules, Higher Quality Products, More Successful Projects, Enhanced Careers*. 1st edition. Addison Wesley, 2003. 272 s. ISBN 0-321-19367-9.
189. MCCOY, D., et al. 2003a. *Gartner 360°: Application Integration and Middleware*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
190. MCCOY, D., et al. 2003b. *Hype Cycle for Application Integration and Platform Middleware 2003*. Gartner, May 2003.
191. MCGOVERN, F. 2003. *Managing Business Process for EAI.* Business Integration Journal [online], 2003, September. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
192. MCGREGOR, J. D., SPIKES, D. A. 2001. *A Practical Guide To Testing Object-Oriented Software*. 1st edition. Addison Wesley, 2001. 417 s. ISBN 0-201-32564-0.
193. MDC. 1999. *Open Information Model* [online]. Meta Data Coalition, 1999. Dostupný na WWW: <http://xml.coverpages.org/MDCOIM11-pdf.gz>.
194. MDC. 2000. *MDC Open Information Model (OIM) Technology Report* [online]. Cover Pages, 2000. Dostupný na WWW: <http://xml.coverpages.org/mdc-oim.html>.
195. META. 2003. *E-Business Evaluations Enterprise Portals - 2002/2003*. META Group, 2003.
196. MICROSOFT. 2005. *Understanding Microsoft Integration Technologies - A Guide to Choosing a Solution*. Microsoft, 2005. Dostupný na WWW: <http://msdn.microsoft.com/library/default.asp? url=/library/en-us/bts_2004wp/html/14bc36a8-69a9-48ed-8e4c-1c85202544c0.asp>.
197. MIERS, D., HARMON, P. 2005. *The 2005 BPM Suites Report 1.0*. Business Process Trends, 2005. 179 s. Dostupný na WWW: <http://www.bptrends.com>.
198. MODI, R., MISHRA, N. 2003. *Are Web Services an alternative to EAI?*. SETLabs Briefings [online], Infosys, 2003, vol. 1, no. 2 (Integration Strategies), s. 15-26. Dostupný na WWW: <http://www.infosys.com/technology/toc_integrationstrategies.asp>.
199. MORAGENTHAL, JP. 2005a. *Enterprise Information Integration: A Pragmatic Approach*. 1st edition. Lulu Press, 2005. 324 s. ISBN 1411629744.
200. MORAGENTHAL, JP. 2005b. *Metadata: A Key to EAI.* Business Integration Journal [online], 2005. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
201. MOSAWI, A.A., ZHAO, L., MACAULAY, L. 2006. *A Model Driven Architecture for Enterprise Application Integration (University of Manchester)*. In 39th Annual Hawaii International Conference on System Sciences (HICSS-38) (IEEE), Hawaii, 2006. Dostupný na WWW: <http://doi.ieeecomputersociety.org/10.1109/HICSS.2006.18>.
202. MOUGIN, P., BARRIOLADE, Ch. 2001. *Web Services, Business Objects and Component Models*. Orchestra Networks, 2001.
203. MURPHY, J., HIGGS, L., QUIRK, C. 2002. *The Portal Framework: The New Battle for the Enterprise Desktop*. AMR Research, 2002. 40 s. AMR-R-10256.
204. NATIS, Y. 2001. *Application Server Scenario: From Stovepipes to Services*. Gartner, October 2001.
205. NATIS, Y. 2002. *Application Server or Application Platform Suite?*. Gartner, October 2002.
206. NATIS, Y. 2003a. *Application Server Scenario: .NET, J2EE and Application Platform Suites*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
207. NATIS, Y., et al. 2003b. *Magic Quadrant for Application Platform Suites: 2Q03*. Gartner, May 2003.
208. NESS. 2004. *Dokumentace k projektu IOM* [interní materiál]. NESS CEE, 2004.
209. NESS. 2006a. *Dokumentace k projektu ADSL SP* [interní materiál]. NESS CEE, 2006.
210. NESS. 2006b. *Dokumentace k projektu SEP* [interní materiál]. NESS CEE, 2006.

211. NEWCOMER, E. 2002. *Understanding Web Services XML, WSDL, SOAP and UDDI*. 1st edition. Addison Wesley, 2002. 216 s. ISBN 0201750813.
212. NEWCOMER, E., LOMOW, G. 2004. *Understanding SOA with Web Services*. 1st edition. Addison Wesley, 2004. 480 s. ISBN 0-321-18086-0.
213. NOCK, C. 2003. *Data Access Patterns: Database Interactions in Object-Oriented Applications*. 1st edition. Addison Wesley, 2003. 512 s. ISBN 0-13-140157-2.
214. NOVOTNÝ, O., POUR, J., SLÁNSKÝ, D. 2005. *Business Intelligence: Jak využít bohatství ve vašich datech*. 1. vydání. Praha: Grada, 2005. ISBN 80-247-1094-3.
215. OBA, M., KOMODA, N. 2001. *Multiple Type Workflow Model for Enterprise Application Integration (Osaka University)*. In 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 7 (IEEE), Hawaii, 2001. Dostupný na WWW: <http://csdl.computer.org/comp/proceedings/ hicss/2001/0981/07/09817048.pdf>.
216. OMG. 2001a. *A UML Profile for Enterprise Distributed Object Computing, Joint Final Submission, Part I, v0.29* [online]. OMG, 2001. Dostupný na WWW: <http://www.omg.org/technology/documents/recent/omg_modeling.htm>.
217. OMG. 2001b. *Common Warehouse Metamodel (CWM) Specification v1.0* [online]. OMG, 2001. Dostupný na WWW: <http://www.omg.org/technology/documents/recent/omg_modeling.htm>.
218. OMG. 2001c. *UML Profile and Interchange Models for Enterprise Application Integration (EAI)* [online]. OMG, 2001. Dostupný na WWW: <http://www.omg.org/technology/documents/recent/omg_modeling.htm>.
219. OMG. 2004a. *Enterprise Collaboration Architecture (ECA) Specification* [online]. OMG, 2004. Dostupný na WWW: <http://www.enterprise-component.com/docs/>.
220. OMG. 2004b. *OMG Unified Modeling Language 2.0 Superstructure* [online]. OMG, 2004. Dostupný na WWW: <http://www.omg.org/cgi-bin/doc?ptc/2004-10-02>.
221. OMG. 2006. *Business Process Modeling Notation (BPMN) Specification*. OMG, 2006. Dostupný na WWW: <http://www.bpmn.org/>.
222. OMG. *Object Management Group "Meta Object Facility"* [online]. OMG. Dostupný na WWW: <http://www.omg.org/technology/documents/modeling_spec_catalog.htm#MOF>.
223. *OpenEAI Project* [online]. Dostupný na WWW: <http://www.openeai.org>.
224. OROVIC, V. 2004. *Modeling for Integration (How much is Enough?).* Business Integration Journal [online], 2004, January. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
225. OVUM. 2001. *Ovum Evaluates Enterprise Application Integration 2001*. Ovum, 2001.
226. PAC. 2003. *EAI and Webservices 2003 Germany Snapshot*. Pierre Audoin Consultants (PAC), 2003. Dostupný na WWW: <http://www.pac-online.com/>.
227. PALMER, N. 2005. *BPM 2005: Market Milestone Report*. Delphi Group, 2005. 100 s.
228. PANCHA, G. 2002. *EAI and ETL: Meeting the Integration Challenge*. eAI Journal [online], 2002, July. Dostupný na WWW: <http://www.eaijournal.com>.
229. PAPAZOGLOU, M. P. 2003b. *Service - Oriented Computing: Concepts, Characteristics and Directions*. In 4th International Conference on Web Information Systems Engineering (WISE'03), 2003. Dostupný na WWW: <http://csdl2.computer.org/persagen/DLAbsToc.jsp?resourcePath=/dl/proceedings/&toc=comp/proceedings/wise/2003/1999/00/1999toc.xml&DOI=10.1109/WISE.2003.1254461>. DOI 10.1109/WISE.2003.1254461.
230. PAPAZOGLOU, M. P., HEUVEL, van den, W. J. 2005. *Service Oriented Architectures: Approaches, Technologies and Research Issues*; The VLDB Journal [online], 2005. Dostupný na WWW: <http://infolab.uvt.nl/pub/papazogloump-2005-81.pdf>. ISSN 1066-8888.
231. PAPAZOGLOU, M. P., KRAMER, B. J., YANG, J. 2003a. *Leveraging Web-Services and Peer-to-Peer Networks*. In 15th Conference On Advanced Information Systems Engineering (CAiSE 2003), Austria, 2003. Springer, 2003, s. 485–501. LNCS 2681.

232. PEPER, I. *Enterprise Application Integration*, Seminární práce, Technische Universität Braunschweig, Institut für Betriebssysteme und Rechnerverbund. Matrikelnummer 2 56 41 77.
233. PERLSTEIN, L. 2003. *Web Services Standards: De Facto, De Jure or Defunct?*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
234. PEZZINI, M. 2003. *Leveraging Legacy Assets: Web Services Through Integration*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
235. PHIFER, G. 2003. *The Portal Is Dead, Long Live the Portal*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
236. PHIFER, G., et al. 2004. *Magic Quadrant for Horizontal Portal Products 2004.* Gartner, 2004.
237. PINKSTON, J. 2001. *The Ins and Outs of Integration - How EAI Differs from B2B Integration*. eAI Journal [online], 2001, August. Dostupný na WWW: <http://www.eaijournal.com>.
238. PLUMMER, D. 2003a. *SODA Is Application Integration for Developers*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
239. PLUMMER, D., DRIVER, M., SMITH, D. 2003b. *The Integrated Service Environment Market Magic Quadrant*. Gartner, April 2003.
240. POOL, K. 2003. *Maintaining & Enhancing Large-Scale EAI.* Business Integration Journal [online], 2003, November. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
241. PŘÍVOZNÍK, L. 2004. *Problematika integrace aplikací při implementaci rozsáhlého informačního systému v utilitních společnostech*. In System Integration 2004. Praha : VŠE KIT, 2004, s. 151-160. ISBN 80-245-0701-3.
242. PUSCHMANN, T., ALT, R. 2004. *Process Portals – Architecture and Integration (University of St. Gallen)*. In 37th Annual Hawaii International Conference on System Sciences (HICSS-04) (IEEE), Hawaii, 2004. Dostupný na WWW: <http://csdl.computer.org/comp/proceedings/ hicss/2004/2056/08/205680225c.pdf>.
243. PYROVOLAKIS, O. I., GARBI, A., PLATANIOTIS, A. 2003. *Web-Service Framework for Business Process Modelling & Legacy Systems Integration*. eChallenges, Bologna, 2003. Dostupný na WWW: <http://citeseer.csail.mit.edu/pyrovolakis03webservice.html>.
244. QUARTEL, D., DIJKMAN, R., SINDEREN, van, M. 2004. *Methodological Support for Service-oriented Design with ISDL (University of Twente)*. In 2nd International Conference on Service Oriented Computing, New York, 2004. ACM Press, 2004. ISBN 1-58113-871-7.
245. RAMAN, V., et al. 2003. *Services for Data Access and Data Processing on Grids* [online]. Global Grid Forum, 2003. Dostupný na WWW: <http://www.gridforum.org>.
246. RAMOS, L. 2003a. *Portal Vendor Landscape: 2003 and Beyond.* Giga Information Group, 2003. 14 s.
247. RAMOS, L., ROOT, N. 2003b. *IT Trends 2004: Enterprise Portals*. Giga Information Group 2003.
248. *ReadySET Pro: Enterprise-Ready Software Engineering Templates*. Dostupný na WWW: <http://www.readysetpro.com/>.
249. REILLY, J. P., GAUTHIER, P. 2004. *Core Business Entities Model White Paper 1.2* [online]. OSS through Java Initiative, 2004. Dostupný na WWW: <www.ossj.org>.
250. ROFAIL, A. 2005. *Dynamic Business Rules Management & Composite Rules: The Next Generation for Business Logic.* Business Integration Journal [online], 2005, June. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
251. ROGERS, S., KING, D. 2003. *6 EAI Problems You Can Prevent.* Business Integration Journal [online], 2003, December. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.

252. ROCH, E. 2004. *A Service-Oriented Architecture for EAI Quality Assurance.* Business Integration Journal [online], 2004, July. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
253. ROSENBLOOM, S. 2004. *When It Comes to Portals, Legacy Information Is Critical.* Business Integration Journal [online], 2004, February. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
254. ROSS, R. G. 2003. *Principles of the Business Rule Approach*. 1st edition. Addison Wesley, 2003. 352 s. ISBN 0-201-78893-4.
255. ROWE, T. 2000. *Making the Business Case for Middleware*. Business Integration Journal [online], 2000, January. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
256. RUBIOLO, D., et al. 2005. *Přehled architektury .NET*. Microsoft, 2005, 146 s.
257. RYMER, J. R., GILPIN, M., VOLLMER, K. 2004. *Integration Landscape*. Forrester Research, 2004.
258. ŘEPA, V. 2006a. *Podnikové procesy - Procesní řízení a modelování*. 1. vydání. Praha: Grada, 2006. 268s. ISBN 80-247-1281-4.
259. ŘEPA V., et al. 2006b. *Metodikou řízené modelování*. Konference Sybase, 2006. Dostupný na WWW <http://sybase.25228.com/prezentace_metodiky.zip>.
260. SEACORD, R. C., PLAKOSH, D., LEWIS, G. A. 2003. *Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices*. 1st edition. Addison Wesley, 2003. 352 s. ISBN 0-321-11884-7.
261. SEI. *Technology Descriptions* [online]. Carnegie Mellon Software Engineering Institute. Dostupný na WWW: <http://www.sei.cmu.edu/str/descriptions>.
262. SESSIONS, R., SICKLER, J. V. 2003. *Software Fortresses: Modeling Enterprise Architectures*. 1st edition. Addison Wesley, 2003. 304 s. ISBN 0-321-16608-6.
263. SHALLOWAY, A., TROTT, J. R. 2004. *Design Patterns Explained: New Perspective on Object-Oriented Design*. 2nd edition. Addison-Wesley, 2004. 480 s. ISBN 0321247140.
264. SHANNON, R. 2005. *Inline Frames* [online]. 2005. Dostupný na WWW: <http://www.yourhtmlsource.com/frames/>.
265. SHARMA, R., STEARNS, B., NG, T. 2001. *J2EE Connector Architecture and Enterprise Application Integration*. 1st edition. Addison Wesley, 2001. 416 s. ISBN 0-201-75580-8.
266. SCHELP, J., SCHWINN, A. 2005. *Extending the Business Engineering Framework for Application Integration Purposes (University of St. Gallen)*. In ACM Symposium on Applied Computing, 2005, str. 1333-1337.
267. SCHMELZER, R. 2003. *Solving Information Integration Challenges in a Service Oriented Enterprise*. Zapthink, 2003. Dostupný na WWW: <http://www.zapthink.com>.
268. SCHMIDT, J. 2002a. *EAI Principles - Part 1*. eAI Journal [online], 2002, October. Dostupný na WWW: <http://www.eaijournal.com>.
269. SCHMIDT, J. 2002b. *EAI Principles - Part 3*. eAI Journal [online], 2002, December. Dostupný na WWW: <http://www.eaijournal.com>.
270. SCHMIDT, J. 2003a. *EAI Lifecycle Evaluation*. eAI Journal [online], 2003, April. Dostupný na WWW: <http://www.eaijournal.com>.
271. SCHMIDT, J. 2003b. *EAI Methodology - The Theory of Application Integration* [online]. Integration Consortium, 2003. Dostupný na WWW: <http://www.wwintegration.com/EAI%20Methodology.pdf>.
272. SCHMIDT, J. 2003c. *EAI Principles - Part 4*. eAI Journal [online], 2003, January. Dostupný na WWW: <http://www.eaijournal.com>.
273. SCHMIDT, J. 2003d. *EAI Principles - Part 5*. eAI Journal [online], 2003, February. Dostupný na WWW: <http://www.eaijournal.com>.
274. SCHMIDT, J. 2004a. *Architecture Revisited.* Business Integration Journal [online], 2004, January. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.

275. SCHMIDT, J. 2004b. *EAI Market Predictions.* Business Integration Journal [online], 2004, March. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
276. SCHMIDT, J. 2004c. *Global Integration Framework* [online]. Integration Consortium, 2004. Dostupný na WWW: <http://www.integrationconsortium.org/page.php?page_id=93&parentId=6>
277. SCHULTE, R. 2005. W. *Magic Quadrant for Integration Backbone Software 1H05*. Gartner, 2005.
278. SCHULTE, R., THOMPSON, J. 2004. *The Enterprise Service Bus in Your Future*. Gartner, October 2004.
279. SCHULTE, R. 2003. *Application Integration Scenario: Making It All Work Together*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
280. SIFTER, Ch. J. 2003. *Integration Reusability*. Business Integration Journal [online], 2003, August. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
281. SINGLETARY, L. A. 2002. *Empirical Study of Stakeholders Perceived Benefits of Integration Attributes for Enterprise IT Applications (Louisiana State University).* In Eighth Americas Conference on Information Systems, 2002. Dostupný na WWW: <www.cab.latech.edu/~lsingle/ homepage/Papers/LAS_AMCIS02_Integration.pdf>.
282. SKOGLUND, S. 2002. *Extending Life Expectancy of Existing Enterprise Information Systems - A Study within SKF AB* [online]. Göteborg University, Department of Informatics, 2002. Dostupný na WWW: <http://www.handels.gu.se/epc/archive/00001484/>.
283. SMITH, D., et al. 2002. *Enterprise Integration*. The Architect [online], 2002, vol. 5, no. 4. Dostupný na WWW: <http://www.sei.cmu.edu/news-at-sei/columns/the_architect/2002/4q02/architect-4q02.pdf>.
284. *SOA Blueprints* [online]. Dostupný na WWW: <http://www.soablueprints.com>.
285. SOPHOS, A. 2005. *Observations from a Tech Architect: Enterprise Implementation Issues & Solutions / Testing* [online]. 2005. Dostupný na WWW: <http://eai.ittoolbox.com/blogs/featuredentry.asp ?i=3071#trackback>.
286. SRIVASTAVA, M. 2003. *Legacy Integration: Which approach should your enterprise adopt?*. SETLabs Briefings [online], Infosys, 2003, vol. 1, no. 2 (Integration Strategies), s. 39-44. Dostupný na WWW: <http://www.infosys.com/technology/toc_integrationstrategies.asp>.
287. STOKES, N. 2001. *EAI and Beyond: A Multi-Level Flow Model A Multi-Level Flow Model*. eAI Journal [online], 2001, May. Dostupný na WWW: <http://www.eaijournal.com>.
288. STOKES, N. *EAI and Beyond: A Multilevel Flow Model*. DataMirror Corporation. Dostupný na WWW: <http://eai.ittoolbox.com/white-papers/eai-and-beyond-a-multilevel-flow-model-1469>.
289. STRÜVER, S. C. 2002. *The Impact of Web Services in the context of Enterprise Application Integration in the Financial Services Industry*. Diplomová práce, Competence Center EAI, Technische Univerzität Berlin, 2002. 127 s. Matrikelnummer 171443.
290. STUMPF, J. 2004. *Integrace aplikací využitím podnikové sběrnice služeb (Enterprise Service Bus)*. In System Integration 2004. Praha : VŠE KIT, 2004, s. 127–134. ISBN 80-245-0701-3.
291. SULLIVAN, D. 2003. *Proven Portals: Best Practices for Planning, Designing, and Developing Enterprise Portals*. 1st edition. Addison Wesley, 2003. 224 s. ISBN 0-321-12520-7.
292. SUN. *Enterprise Application Integration*. Sun Microsystems. Dostupný na WWW: <http://java.sun.com/blueprints/guidelines/designing_webservices/integration.pdf>.
293. SVÁTEK, V., VÁVRA, V. 2004. *Sémantická integrace webových služeb*. In System Integration 2004. Praha : VŠE KIT, 2004, s. 253-260. ISBN 80-245-0701-3.

294. TAYLOR, J. 2004. *Managing Information Technology Projects: Applying Project Management Strategies to Software, Hardware, and Integration Initiatives*. 1st edition. Amacom, 2004. 274 s. ISBN 0814408117.
295. TEILHARD. 2003. *Strategic Market Valuation Report*. Teilhard Technologies, 2003. 73 s.
296. THE OPEN GROUP. 2003. *TOGAF v8.1*. The Open Group, 2003. Dostupný na WWW: <http://www.opengroup.org/architecture/togaf8/index8.htm>.
297. THEMISTOCLEOUS, M., et al. 2001a. *ERP Problems and Application Integration Issues: An Empirical Survey (Brunel University)*. In 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 9 (IEEE), Hawaii, 2001. Dostupný na WWW: <http://www.hicss.hawaii.edu/HICSS_34/PDFs/ST2EA03.pdf>.
298. THEMISTOCLEOUS, M., et al. 2004. *Extending the Information System Lifecycle through Enterprise Application Integration: A Case Study Experience (Brunel University)*. 37th Annual Hawaii International Conference on System Sciences (HICSS-04) (IEEE), Hawaii, 2004. Dostupný na WWW: <http://csdl.computer.org/comp/proceedings/ hicss/2004/2056/08/205680228b.pdf>.
299. THEMISTOCLEOUS, M., IRANI, Z. 2003. *Integrating Cross-Enterprise Systems: An Innovative Framework for the Introduction of Enterprise Application Integration*. In 11th European Conference on Information Systems (ECIS 2003), 2003. Dostupný na WWW: <http://csrc.lse.ac.uk/asp/aspecis/20030160.pdf>.
300. THEMISTOCLEOUS, M., IRANI, Z. 2006. *Towards a Methodology for the Development of Integrated IT Infrastructures (Brunel University)*. In 39th Annual Hawaii International Conference on System Sciences (HICSS-39) (IEEE), Hawaii, 2006. Dostupný na WWW: <http://doi.ieeecomputersociety.org/10.1109/HICSS.2006.492>.
301. THEMISTOCLEOUS, M., IRANI, Z., O'KEEFE, R. M. 2001b. *ERP and application integration: Exploratory survey.* Business Process Management Journal, MCB University Press, 2001, vol. 7, no. 3, s. 195-204. ISSN 1463-7154.
302. THOMAS, N., BUCKLEY, W. *The Enterprise Service Bus*. Web Services Journal [online], Dostupný na WWW: <http://www.sys-con.com/webservices/article.cfm?id=668&count=15703&tot= 4&page=1>.
303. TIBCO. 2002. *Universal Application Network and TIBCO*. Tibco, 2002. Dostupný na WWW: <www.tibco.com/resources/software/ standards_support/uan_whitepaper.pdf>.
304. TIBCO. 2003a. *TIBCO InConcert™ Concepts 6.2*. Tibco, 2003. Dostupný na WWW: <http://www.tibco.com>.
305. TIBCO. 2003b. *TIBCO Rendezvous™ Concepts 7.2*. Tibco, 2003. Dostupný na WWW: <http://www.tibco.com>.
306. TIBCO. 2004. *TIBCO® BusinessWorks Workflow Concepts 5.1.0*. Tibco, 2004. Dostupný na WWW: <http://www.tibco.com>.
307. TIBCO. 2005. *Process Rules:Getting More from Business Process Management with a Business Rules Engine*. Tibco, 2005. Dostupný na WWW: <http://www.tibco.com>.
308. TIDWELL, D., SNELL, J., KULCHENKO, P. 2001. *Programming Web services with SOAP*. 1st edition. O'Reilly, 2001. 216 s. ISBN 0-596-00095-2.
309. TMF. 2003. *Shared Information/Data (SID) Model, GB922, v3.1* [online]. Telemanagement Forum, 2003. Dostupný na WWW: <http://www.tmforum.org>.
310. *TOP10 Systémových integrátorů* [online]. Dostupný na WWW: <http://top10.hottop.cz/php/>.
311. TOTLANI, R. 2004. *Exception-Handling Models in EAI.* Business Integration Journal [online], 2004, July. Dostupný na WWW: <http://www.bijonline.com>. ISSN 1552-2326.
312. TOUSSAINT, A. 2005a. *BEA WebLogic Portal™ Roadmap*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.
313. TOUSSAINT, A., BAUMGART, T. 2005b. *Federated Portals*. Slidy z konference BEA World 2005. Dostupný na WWW: <http://beaworldonline.bea.com>.

314. TREADWELL, J. 2005. *Open Grid Services Architecture: Glossary of Terms* [online]. Global Grid Forum, 2005. Dostupný na WWW: <http://www.gridforum.org>.
315. TSE, W. *Enterprise Application Integration* (materiály k přednáškám) [online]. UCL Department of Computer Science. Dostupný na WWW: <http://www.cs.ucl.ac.uk/staff/W.Emmerich/lectures/3C05-03-04/EAI.pdf>.
316. TUECKE, S., et al. 2003. *Open Grid Services Infrastructure (OGSI) v1.0* [online]. Global Grid Forum, 2003. Dostupný na WWW: <http://www.gridforum.org>.
317. VALDES, R. 2003. *Enterprises Face Outward With Portals and Web Services*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
318. VARLAMOV, S. 2002. *Security Strategies for EAI*. eAI Journal [online], 2002, September. Dostupný na WWW: <http://www.eaijournal.com>.
319. VASCONCELOS, A., et al. 2004. *An Information System Architectural Framework for Enterprise Application Integration*. In 37th International Conference on System Sciences, Hawaii, 2004. Dostupný na WWW: <http://csdl.computer.org/comp/proceedings/ hicss/2004/2056/08/205680225b.pdf>.
320. VECCHIO, D. 2003a. *Magic Quadrant for Programmatic Integration Servers 2003*. Gartner, December 2003.
321. VECCHIO, D. 2003b. *Service-Oriented Architecture for Legacy: Junkyard Wars for Application Services*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
322. VOLLMER, K., PEYRET, H. 2005. *The Forrester Wave: Integration Suites: Q3 2005*. Forrester Research, 2005.
323. VONDRÁK, I. 2004. [online]. *Metody byznys modelování pro kombinované a distanční studium*. 1. vydání. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, Fakulta elektrotechniky a informatiky, 2004. 92 s. Dostupný na WWW: <http://vondrak.cs.vsb.cz/download.html>
324. VOŘÍŠEK, J. 1997. *Strategické řízení informačního systému a systémová integrace*. 1.vydání. Praha: Management Press, 1997. 324 s. ISBN 80–85943–40–9.
325. VOŘÍŠEK, J. *Materiály k přednáškám kurzu Informační systémy a technologie (IT_215)* [online]. Dostupný na WWW: <http://nb.vse.cz/~vorisek/FILES/IT215_materialkpredmetu>.
326. VOŘÍŠEK, J., et al. 2004. *Aplikační služby IS/ICT formou ASP: Proč a jak pronajímat informatické služby*. 1.vydání. Praha: Grada, 2004. 213 s. ISBN 80–247-0620–2.
327. W3C. 2004. *XML Query* [online]. W3C, 2004. Dostupný na WWW: <http://www.w3.org/XML/Query>.
328. WAGNER, R. 2003. *Security for the Web-Services-Enabled Enterprise*. Gartner Symposium ITxpo 2003, Gartner, March 2003.
329. WAKE, C. W. 2001. *Extreme Programming Explored*. 1st edition. Addison Wesley, 2001. 144 s. ISBN 0201733978.
330. WHATIS. *Definitions, Dictionary for Internet & Computer Technologies* [online]. WhatIs.com. Dostupný na WWW: <http://www.whatis.com>.
331. *Wikipedia, the free encyclopedia* [online]. Dostupný na WWW: <http://www.wikipedia.org>.
332. WOODGATE, S., et al. 2004. *Microsoft BizTalk Server 2004 UNLEASHED*. 1st edition. Sams Publishing, 2004. 768 s. ISBN 0-672-32598-5.
333. WSDIS. 2004. *Information Service Board - Core System Framework - Business Justification Outline* [online]. Washington State Department of Information Services, 2004. Dostupný na WWW: <http://dis.wa.gov/isb/coresystem/justification.htm>.
334. YACOUB, S. M., AMMAR, H. H. 2003. *Pattern-Oriented Analysis and Design: Composing Patterns to Design Software Systems*. 1st edition. Addison Wesley, 2003. 416 s. ISBN 0-201-77640-5.

Poznámka: Časopis eAI Journal byl v roce 2005 sloučen s časopisem Business Integration Journal, proto jsou v současné době články z časopisu eAI Journal online nedostupné.