University of Economics, Prague

Faculty of Informatics and Statistics

Department of Information Technologies

Student: **Peter Fedoročko**

Supervisor of bachelor thesis: **doc. Ing. Ján Pour CSc.**

OPTIMIZATION OF COSTS AND PRODUCT PROFITABILITY USING WHAT-IF ANALYSIS IN RECIPROCAL COST ALLOCATION MODEL

YEAR: 2010

Enouncement

I honestly declare that I have prepared my bachelor thesis by myself and that I have adduced all resources and literature I used.

Prague, 30.4.2010 .....

Acknowledgement

I would like to thank my supervisor, doc. Ing. Jan Pour, CSc. for his continual support and advice throughout this work and my colleagues from IBM for many helpful reflections.

Abstrakt

Práca je zameraná na jednu z najdiskutovanejších oblastí manažérskych informačných systémov súčasnosti – Corporate Performance Management. Nesnaží sa však o detailný výklad danej témy, naopak orientuje sa na špecifickú problematiku integrácie jedného z kľúčových riešení nákladového a manažérskeho účtovníctva – nákladovej alokácie, do prostredia nástrojov Performance Management.

Práca je rozdelená na teoretickú a praktickú časť. Prvá časť je venovaná vymedzeniu termínu Corporate Performance Management, jeho umiestneniu v architektúre Business Intelligence nástrojov ako aj identifikácii dôvodov pre jeho implementáciu s ohľadom na súčasnú ekonomickú situáciu. Z troch základných pilierov Performance Management je najväčšia pozornosť venovaná analytickej časti umožňujúcej vykonávanie tzv. what-if analýz. Nákladovej alokácií je venovaná časť identifikujúca hlavné prínosy jej implementácie v podobe optimalizácie a minimalizácie výdajov rovnako ako zdôvodnenie jej nasadenia v bankovom sektore.

Praktická časť dokumentuje jednotlivé fázy projektu implementácie analytického riešenia nákladovej alokácie vo fiktívnej banke, pričom kľúčová časť je venovaná hlavne architektúre a návrhu riešenia výpočtového jadra. Záver práce popisuje implementáciu celého riešenia v nástroji IBM Cognos TM1, pričom doraz je kladený na využitie pokročilej funkcionality a vysvetlenie implementačných metód.

Kľúčové slová: Corporate Performance Management, What-if analýzy, Nákladová alokácia, IBM Cognos TM1

Abstract

Thesis is focused on one of the most discussed area of Management Information Systems currently – Corporate Performance Management. It does not aim on detail explanation of theme itself, but orientate to specific problematic of setting key solution of managerial accounting – cost allocation, to the Performance Management systems.

Thesis is divided into theoretical and practical part. First part is devoted to Corporate Performance Management definition, its placement in architecture of Business Intelligence devices as well as reasoning for its implementation considering current economical situation. From the main three pillars of Performance Management is major attention dedicated to analytical part enabling execution of what-if analyses. Part devoted to cost allocation identifies contributions of its implementation in optimization and minimization of expenditures as well as offers reasons for using them in bank sphere.

Practical case study documents particular project phases of implementation of cost allocation analytical solution in fictive bank, while major section is dedicated to architecture and design part of the calculation engine. Project is concluded with description and explanation of advanced implementation techniques and methods in IBM Cognos TM1 software.

Key words: Corporate Performance Management, What-if analysis, Cost Allocation, IBM Cognos TM1

Table of Content

I. Introduction ..... - 9 -

II. Theoretical scope ..... - 10 -

1. Performance management ..... - 10 -
   1.1. Answers around us ..... - 10 -
   1.2. Performance Management as super-structure of Business Intelligence ..... - 11 -
      1.2.1. Vertical slice ..... - 11 -
      1.2.2. Horizontal slice ..... - 12 -
         1.2.2.1. Reporting ..... - 12 -
         1.2.2.2. Planning ..... - 12 -
         1.2.2.3. What – if analysis ..... - 13 -
   1.3. Reasons for moving from BI to CPM ..... - 14 -
2. What-if analysis ..... - 15 -
   2.1. Definition ..... - 15 -
   2.2. Basic requirements and recommendations for what-if analytical system ..... - 15 -
      2.2.1. Technical recommendations ..... - 16 -
         2.2.1.1. Proper technology for write back capabilities ..... - 16 -
      2.2.2. Business requirements ..... - 17 -
   2.3. What-if analysis conclusion ..... - 18 -
3. Performance Management's versatility ..... - 18 -
4. Cost allocation ..... - 20 -
   4.1. Simple example ..... - 20 -
   4.2. Theoretical summing-up ..... - 23 -
      4.2.1. Types of costs ..... - 23 -
         4.2.1.1. Variable costs ..... - 23 -
         4.2.1.2. Fixed costs ..... - 23 -
      4.2.2. Cost Drivers ..... - 23 -
      4.2.3. Drivers Assignment ..... - 24 -
      4.2.4. Types of cost centers ..... - 24 -
         4.2.4.1. Production centers ..... - 25 -
         4.2.4.2. Support centers ..... - 25 -
         4.2.4.3. Reasons for dividing cost centers in cost allocation ..... - 25 -
      4.2.5. Types of cost allocation ..... - 25 -
         4.2.5.1. Simple allocation ..... - 26 -
         4.2.5.2. Waterfall (cascade) allocation ..... - 27 -
         4.2.5.3. Reciprocal allocation ..... - 27 -
      4.2.6. Conclusion of theoretical summing-up ..... - 27 -

III. Case study ..... **- 29 -**
5. Introduction to the case study ..... - 29 -
6. Order specifications and requirements ..... - 29 -
7. Selected technology ..... - 30 -
8. Project solution ..... - 30 -
8.1. Analyze of input data sources ..... - 31 -
8.1.1. Outputs of processes ..... - 32 -
8.1.2. Other sources ..... - 33 -
8.2. Architecture and design of the cubes ..... - 33 -
8.3. Implementation ..... - 42 -
8.3.1. Introduction of TM1 environment ..... - 42 -
8.3.2. Revenue component ..... - 43 -
8.3.2.1. Product dimension ..... - 43 -
8.3.2.2. Time dimension ..... - 45 -
8.3.2.3. Version dimension ..... - 45 -
8.3.2.4. Sales metrics dimension ..... - 45 -
8.3.2.5. Prices cube ..... - 46 -
8.3.2.6. Sales&Production Cube ..... - 48 -
8.3.3. Variable costs component ..... - 52 -
8.3.4. Fixed costs allocation component ..... - 53 -
8.3.4.1. CC&P Dimension ..... - 53 -
8.3.4.2. Driver's Name cube ..... - 54 -
8.3.4.3. Driver's Value dimension ..... - 54 -
8.3.4.4. Drivers Package ..... - 56 -
8.3.4.5. Drivers Assignment ..... - 56 -
8.3.4.6. Cost Allocation_Absolute ..... - 57 -
8.3.4.7. Cost Allocation ..... - 58 -
8.3.4.8. Amounts for Allocation ..... - 58 -
8.3.4.9. Multiplied ..... - 59 -
8.3.4.10. Allocated Costs on Product ..... - 60 -
8.3.4.11. Manager Console for Cost Allocation ..... - 61 -
8.3.4.12. Allocate process ..... - 61 -
8.3.4.13. Repeat X Times process ..... - 62 -
8.3.4.14. Products Profitability ..... - 65 -
8.4. Development of User Interface ..... - 66 -
8.5. Exemplary what-if analysis ..... - 68 -
IV. Findings ..... **- 70 -**
V. Dictionary ..... **- 71 -**
VI. Literature ..... **- 72 -**

I. Introduction

Nowadays, more and more companies solve problem of cost minimization and revelation of hidden expenditures. Determination of net profit, on any level of detail could be very difficult especially in sectors where fixed costs are highly above variable. The main feature of fixed costs is that they are not directly interconnected with final output which produces the revenue. In case of majority, they could significantly warp the total amount of resources spend on particular objects and often manage company to loss. This is the reason why I chose problematic of cost allocation as a storey post of my paper. In the following letters I would like to root this purely financial thesis into the Business Intelligence environment. The important accent will be placed on ability to not only observe the aspect, however to optimize results through what-if analytical capabilities.

Although, the concept of Business Intelligence is well-known these days I will briefly draw on the IBM's approach to the problematic and the slice of different advanced data usage accesses. Also the Performance Management and its role in Business Intelligence will be presented.

Further, the basic technical and business requirements for what-if analysis will be defined. It will be followed by an ordinary example of simple cost allocation. There will be explained basic techniques, principles and approaches as well as the results it could provide. At short notice will also be mentioned all three types of cost allocation - from simple one to most difficult - reciprocal.

Second part of the thesis will be dedicated to the example of reciprocal cost allocation implementation in banking sector. The problem will be described vertically, from the bottom part represented by prototypal database to the top, represented by analytical application designed for what-if analysis and easy optimization. From the technical point of view will be gradually solved analysis of the input database, its optimal structure as well as the appropriate data transformation for use in multidimensional cubes. The major part will we be dedicated to the implementation along with the business discussion about the required outputs, its further usage and the way user can manage the whole process. From the business point of view will be solved the advantages and disadvantages of this method in comparison to the simple one.

Different types of data from different sources are inevitable for acquiring sufficient results. The thesis will describe ways of obtaining them and their proper usage.

In conclusion two exemplary what-if analyses will be done to demonstrate the benefits it can provide.

My aim in this thesis will be placed on suggestion of possible solution as well as preparation of model, which could be after required modification implemented in corresponding sectors. I hope that strong reasons mentioned in the first part of the thesis will be sufficient argument for reader to consider implementation of this decision making and results optimizing application.

II. Theoretical scope

1. Performance management

1.1. Answers around us

Turbulence of the current economical environment causes insecurity and fear about further progress all around the world. Each of the big economical crises so far has warned about itself in an advanced. How it is possible that no one has noticed and hasn't been able to respond?

Mankind daily generates eight times more information than stored in all US libraries[1]. Nearly all of them are somehow related to the economical indicators. Think about demographical development in India. By its observation we can easily predict the market evolution, reveal rising opportunities and occasions for future investments as well as labor forces which will be available. What about weather in Brazil? Undoubtedly it has great influence on production of particular commodities, which quantities and prices can pretty well waft the marker. Let's move to the even more abstract example. Stochastic curve of the student's attendance can highly influence the university schedule as well as the student's placement to the peculiar classrooms. And what is the financial impact? Education during the specific hours is undoubtedly cheaper than the lectures under the artificial light. Is it not enough? This information could help IT department better predict the optimal structure and capacity of the IT equipment, which is surely not cheap investment. I'm sure we can continue with calculating similar examples. And we even did not mention the huge amount of sentinels that warned us before the last crisis. Indications, answers, right solutions. They are all around us. The only we need is the proper way to collect, store and harvest true value from them.

Each company appears on the outside as an independent entity producing and collecting significant volume of data. On the other hand it is always part of the higher solidus influenced by many miscellaneous factors. These, when used in appropriate way, are able to immediately give answers about performance of the company, how it is doing right now and what it should do to have better results, from the inner as well as from the whole market view. *Use them in appropriate way* will be meant in this text to transmit them to the "hands" of Business Intelligence.

Chuck Ballard and Daniel M. Farrell described in their publication[2] the necessity for companies to explore data as follows:

„They need the ability to transform this raw data to actionable information by capturing, consolidating, organizing, storing, distributing, analyzing, and providing quick and easy access to it. This is the competitive advantage, but also the challenge. All of this is the goal of business intelligence (BI). BI helps a company create knowledge from that information to enable better decision making and to convert those decisions into action."

[1] IBM Smarter Planet – New Intelligence

[2] Ballard Ch., Farrell D., Gupta A., Mazuela C., Vohnik S.: Dimensional Modeling: In a Business Intelligence Environment, page 23

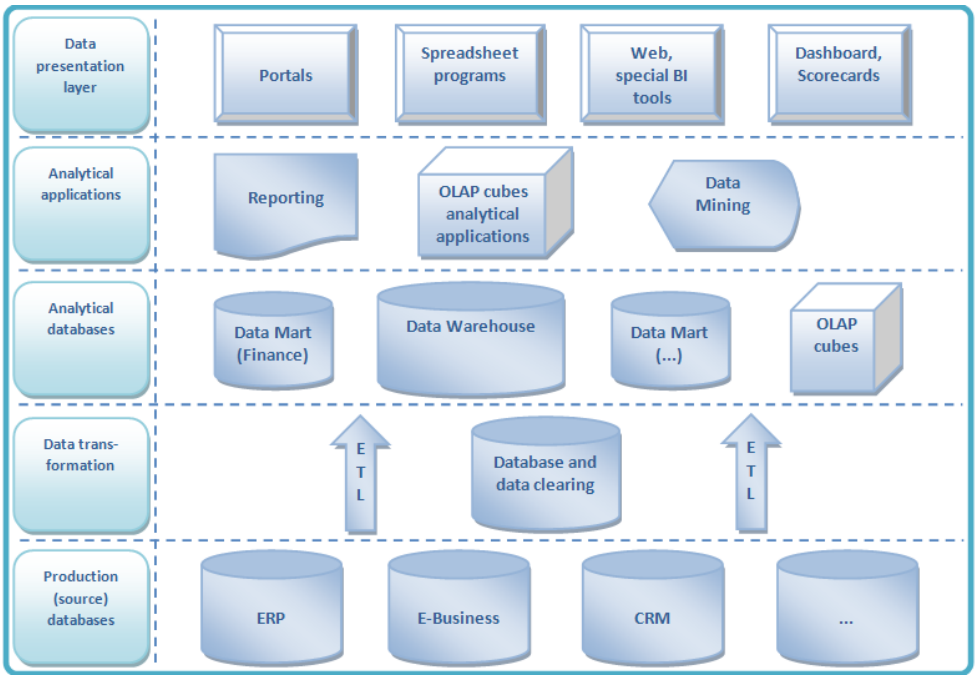
This conversion of decisions into action will play the most important part in optimization of performance. However, it could have more approaches. In the following paragraph I will gradually describe single of them and extract one, which will be the most appropriate for solved problematic.

1.2. Performance Management as super-structure of Business Intelligence

For the better understanding of the Business Intelligence problematic as well as the integrity of the various layers and systems used, two different slices across the BI will be identify further.

1.2.1. Vertical slice

Practically there exists number of different variants and solutions of Business Intelligence, which use and combine various BI technologies and products. In general, every BI solution is based and built on some layers, in which data are handed from original (primary) systems through different stages to the presentation layer where results are generalized to the managers and analysts[3]. Special view on data, from different considerations, dimensions and their combinations requires special storage of input data. Therefore multidimensional approach is used while working with data on analytical layers. Differences between the OLTP and OLAP depositing, as well as the advantages and disadvantages of the particular techniques are well known and because of the limited coverage of this paper will not be explained or discussed further in the text. Upper layers of the model represent processing and transformations of data stored in OLAP layers. These are then presented and accessed through special applications such as spreadsheet programs, professional reporting systems or web interfaces.



Picture 1 : Vertical slice of Business Intelligence, Pour J., Gála L., Šedivá Z.: Podniková informatika

[3] Pour J., Gála L., Šedivá Z.: Podniková informatika, page 218

1.2.2. Horizontal slice

Horizontal slice is represented by the palette of the Business Intelligence devices and indicates the upcoming trend and the further development of the architecture.

1.2.2.1. Reporting

Formally BI included devices used to present the existing data from the data warehouses. This enabled managers and analysts express observation of the defined indicators known as KPI by using dashboards and scorecards. They provide answer to the question *how we are doing*. In case of suspicious values or data divergences, users were able to drill for a more detailed view, observe why it is like that and reveal hidden causalities by using reports. Therefore part of the Business Intelligence which answers questions *how* and *why* is known as reporting.

1.2.2.2. Planning

Apart from data presentation BI enables movement on the time line to the future and defines values for the predicted time interval known as planning. There is numerous way how to perform planning, however the detail description of particular methods overlaps the coverage of this publication and is not directly related to the topic we will discuss further. Nevertheless, for the complexity, the basis will be introduced:

Automatic:

- According to the last year values

	Total	Jan	Feb	Mar	Apr	May	Jun	Jun	Aug
200x	1000	100	140	180	80	200	100	120	80
	100 %	10 %	14 %	18 %	8 %	20 %	10 %	12 %	8 %
200x +1	1500	150	210	270	120	300	150	180	120

Table 1: Planning according to the last year values

- Overlap with the trend curve

In the third line of the table no. 2, monthly growth of 200x year values is calculated. Second column identifies the average perceptual growth of 200x year. We will apply this value to the values of 200x + 1 year. Last value of 200x was 300, which will be raised by 19 % in January[4] month of 200x + 1 year. According to the trend curve, expected value for January 200x + 1 will be 357. Values for February 200x + 1 will be equal to 19 % growth against the January what is 425[5].

	average	Jan	Feb	Mar	Apr	May	Jun	Jun	...
200x	-	100	120	145	175	210	250	300	...
	19,59 %	-	20 %	20,83 %	20,68 %	20 %	19 %	20 %	16,66 %
200x +1	-	357	425	506	602	...	...	...	...

Table 2: Planning by overlapping with the trend curve

- According to the one or more parameters

In this method we expect 10 % growth in 200x + 1 against the particular months in 200x.

[4] 300 * 1,19 = 357
[5] 357 * 1,19 = 425

	Param.	Jan	Feb	Mar	Apr	May	Jun	Jun	...
200x	-	100	120	145	175	210	250	300	...
200x +1	1,1	110	132	160	192,5	231	275	330	...

Table 3: Planning according to the one or more parameters

- Aliquot distribution

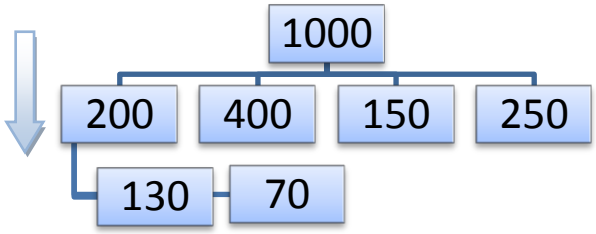
In this method, value from the second column is proportionally divided between twelve months of the 200x year.

	Value	Jan	Feb	Mar	Apr	May	Jun	Jun	...
200x	2400	200	200	200	200	200	200	200	...

Table 4: Planning according to the aliquot distribution

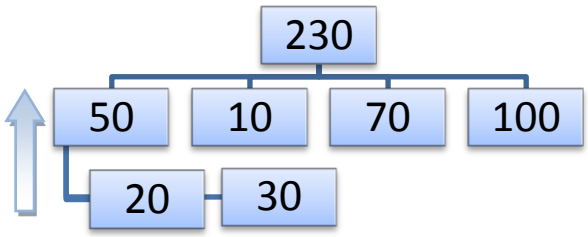
Manual**:**

Top-down – highest element of the structure defines value which has to be divided between children elements. Sum of the values on the children level must be equal parent's element.



Picture 2: Top-down planning

Bottom-up – elements from the lowest level plan values. These are consolidated and construct final value.



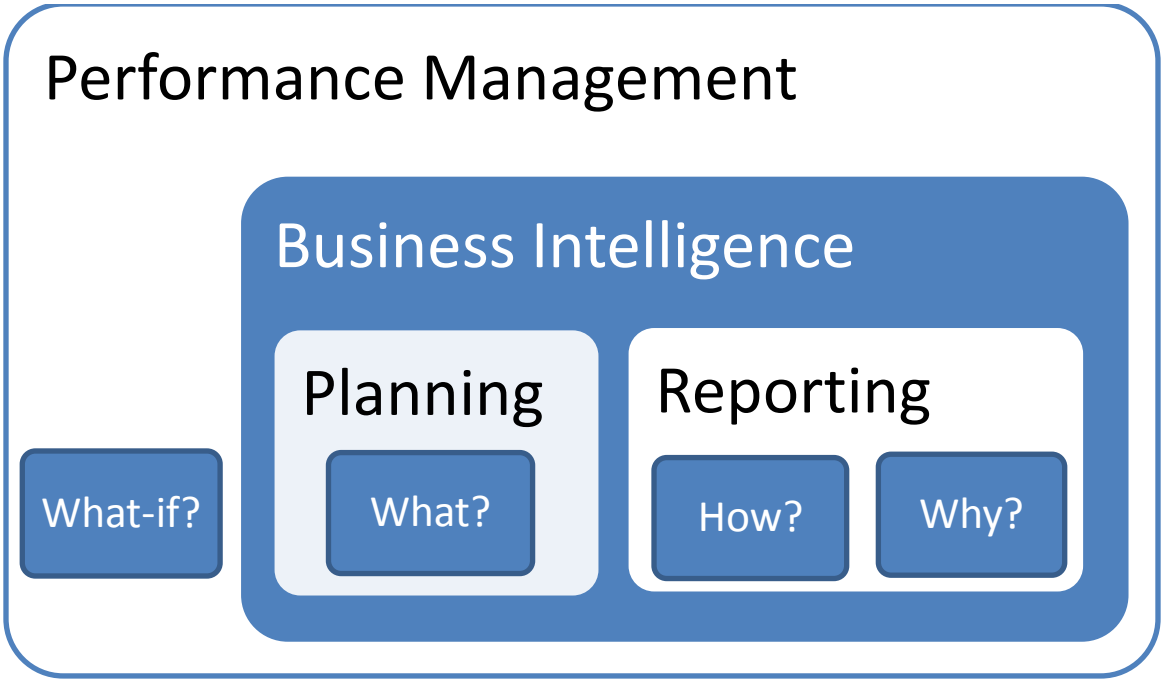
Picture 3: Bottom-up planning

1.2.2.3. What – if analysis

However, solely planning in current economical situation does not have to reflect the reality and future development in any version. Even the smallest factors could enormously influence the evolution and lead to loss. To avoid negative influence or to find an improvement, it is very effective and also very popular these days using analytical planning, better known as what-if analysis. What answer they provide? How would the change in a parameter or group of parameters influence the performance and KPIs of the company? For a better imagination, let's move back to the example from the beginning of the letter. In case that in the model, *demographical development* would be set as the parameter for the purchasing

power of India, we could simply and very easily predict and model different scenarios for the future. Parameters represent weather progress would significantly influence the commodity and stock market. Representatives of university, responsible for the schedule, would have a strong device which could estimate the costs of prepared timetable. Precious and accurate implementation of the model could even suggest changes which would lead to the more appropriate and efficient solution or offer real time calculation of cost while providing *ad-hoc* queries to the model.

After the introduction and detail description of single parts, the graphical representation of Performance Management will be demonstrated. In the next picture, Business Intelligence covers three key questions *How*, *Why* and *What*. The new trend is adding more abstract question *what-if*, which creates and covers complex solution called (Corporate) Performance Management. By combining these four questions, performance of the company could be effectively observed, predicted and managed.



Picture 4: Horizontal slice of Business Intelligence

1.3. Reasons for moving from BI to CPM

Considering the current insecurity on financial markets as well as malefic fiscal situation about the further economical development, we identify as crucial for the company the need for proactively managed performance through CPM[6]. The view on *how we were doing* so far is farther insufficient and companies have to demand for the window to their performance, to the aspects they are possible to challenge in the

[6] Authors of the Business Intelligence (Corporate Performance Management), Ganesh B., Edwards R., Burnett S. and Jennings T. describe the necessity of using CPM during and after the financial instability as follows: *„Economic pressures dictate that organizations manage corporate performance better to adapt to the different world that has emerged from the ashes of the credit crunch."*

future. This could be properly represented by modeling different scenarios, which provide the ability to understand performance information through discovery, analysis, and ad hoc querying[7]. It is the area which is additional to BI and covered exactly by CPM. Therefore, further in this paper, we will focus on driving performance of the company by using *what-if* analysis. *What-if* analyses are often considered to be the indispensible device in the time of crisis and could avoiding destructing consequences as well as identify vital opportunities. Exploring different scenario, discovering hidden relations, analysis trends, executing ad-hoc queries, these are all areas where *what-if* analysis are done and inevitable. However, their performing requires adequate technology as well as special business implementation. In the next paragraph we will take a closer look to this method and its specifications.

2. What-if analysis

2.1. Definition

What-if analysis is representation of real-life system by a model used to determine how a change in one or more variables affects the rest of the system. Every combination of different input values (which were changed) creates unique scenario of the future* which is possible to occur. It provides comparison of alternative behavior under the specific conditions. Selection of the parameters which will be changed as well as the intensity of modification depends on manager or on analyst who execute the analysis. Usually he comes out from acquirement of the analyzed object and situations which are most possible to occur. What-if analysis then enables him to contribute to the knowledge, prepare for the presumable future or identify possible improvement[8].

2.2. Basic requirements and recommendations for what-if analytical system

In previous paragraph, we determined reasons for implementing *Performance Management* system with *what-if* functionality. However, implementation of this "*BI upgrade*" with *what-if* capabilities will require special approach on technical as well as on business side. For better understanding of CPM application architecture as well as best practices for implementation from business point of view and following case study, basic necessary specifications will be described further.

[7] Ganesh B., Edwards R., Burnett S., Jennings T.: Business Intelligence (Corporate Performance Management)

[8] Hnilica J., Fotr J.: Aplikovaná analýza rizika ve finančním managementu a investičním rozhodování, page 57

* It is important to mention that despite many people associate what-if only with future, it could be as well used to modify current condition. Case study in this thesis will closely explain it and show how to enable optimization of present state.

2.2.1. Technical recommendations

Despite the fact, that CPM solution could be realized on many different technologies, some recommendations for obtaining sufficient results and performance will be specified further.

- In-memory system with write back capabilities

  In-memory – Every change of variable in the application (execution of *what-If* analysis) could affect numerous different data and results all over the model. Therefore, for the on-line response to the change, fast access to all data is necessary. This could be assured by loading the whole model into the RAM memory. Data are easily accessible and ever change could be quickly recalculated. On the other hand, large RAM memory is demanded.

  Write-back – This is the basic principle and requirement for *what-if* analysis. Data stored in model are not only for presentation, but thanks to their presence in RAM memory, they can be easily changed with instant impact to the whole model and results which are observed. This capability however requires specific BI architecture and implementation. Two most common architectures of analytical layer will be discussed further as well as reasons why they are or are not appropriate for write-back capabilities.

2.2.1.1. Proper technology for write back capabilities

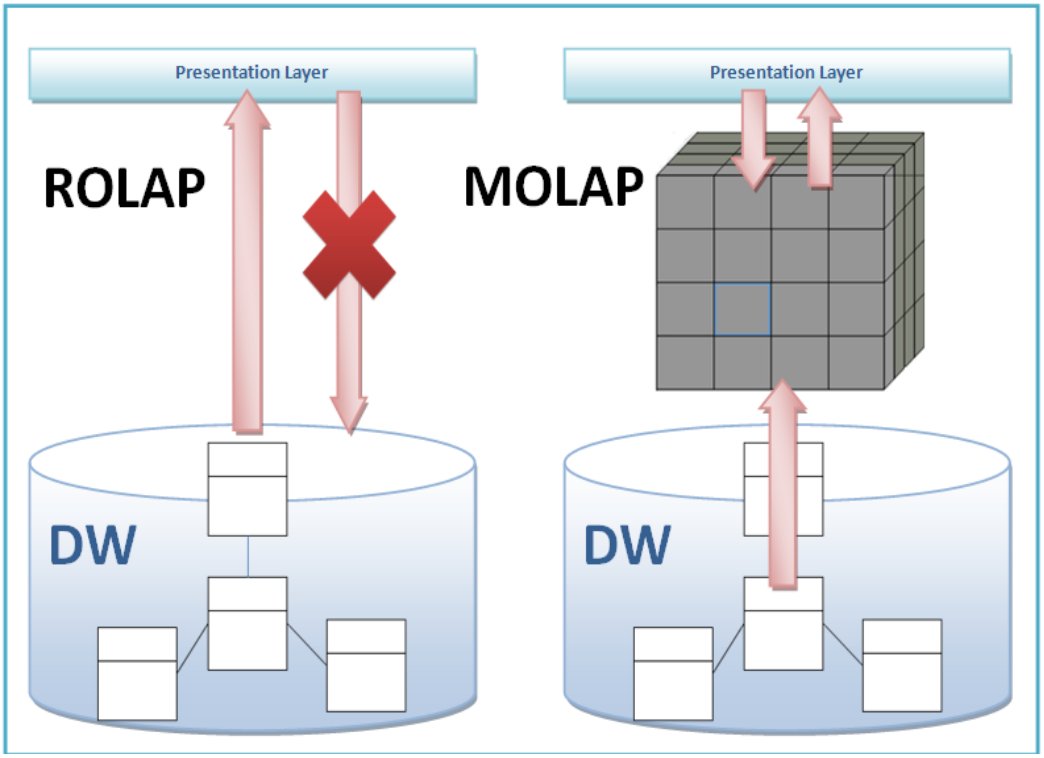
As was shown in the horizontal slice of BI architecture (page 9, picture no.1), data from primary systems are loaded into the analytical applications (represented by analytical layers), where are properly processed and passed to the presentation layers. Further in the text, we will focus on the analytical layers (also known as OLAP), where what-if analyses are executed.

Technology of analytical layers (OLAP) can be realized in some variants. Two most common are ROLAP and MOLAP.

- ROLAP (Relational OLAP) is solution based on multidimensional storage of data in relational databases[9]. Paragraph 1.2.1 shows transformation of *E/R* data from primary systems to the multidimensional storage of data in Data Warehouse. This is appropriate and inevitable for quick creation of required views in BI solutions. As we can see in picture bellow, manager or analyst is accessing data and making queries through presentation layer. He can find out *How* (and *Why)* is his company doing from data stored in Warehouse. However, one of the Data Warehouse characteristic is permanency. Data in DW are *read-only* and cannot be changed through any application[10]. Therefore it is not possible to create what-if analysis and different scenario by changing existing data. Result of this is that ROLAP technology is not appropriate for Performance Management.
- MOLAP (Multidimensional OLAP) is technology with special storage of data in multidimensional – binary cubes[11]. Data from DW are loaded into the cubes by special processes (will be discusses further in practical case study) and then manager or analyst access data directly

[9] Pour J., Gála L., Šedivá Z.: Podniková informatika, page 224
[10] Pour J., Gála L., Šedivá Z.: Podniková informatika, page 227
[11] Pour J., Gála L., Šedivá Z.: Podniková informatika, page 224

from this cubes. Data in cubes are independent from data stored in Warehouses and can be easily changed through the analytical application. Result of this technology is possibility of executing *what-if* analyses. Therefore, MOLAP solution will be used and discussed further in the paper. Reader should consider MOLAP advantages as well as physical independency of data stored in cubes and in DW while reading rest of the paper or implementing his own solution.



Picture 5: Differences between ROLAP and MOLAP technology

2.2.2. Business requirements

After we are familiar with technical requirements for what-if analytical applications, best practices, requirements and recommendations for proper implementation will be discussed. We can identify the most important of them as follows:

- Ambition for the maximum imitation of reality (dimensions, relationships, data-flow)

The more precisely the model describes reality, the better and more accurate results we can expect. During the implementation, it should be well considered and applied. Wrong imitation or relation could even lead to the wrong results and could be followed by incorrect decision.

- Maximum parameterization of the model (parameters counts, abilities to choose from different options)

The more parameters will model have, the more options will manager or analyst have during executing what-if analyses. It enables him to prepare wider palette of situations which are able to occur or be analyzed. Despite, the number of parameters does not have to be injury of lucidity and comprehension of the application for business user.

- User friendly and understandable model (primary dedicated for management and business analysts)
- Linkage of the parameters to all values they actually influence – two-edged paradigm – each parameter will influence some values positively, some negatively (HW investment – performance improvement, response acceleration, on the other hand – cost increase)
  The goal is to find appropriate state, maximization of profit with bearable negative influence and risks. Theory of contradirectional forces describe each progress and expanses as movement suppressed by opposite. This prevent against never ending expanse. Therefore, while building the model, it should be considered and properly implemented.

2.3. What-if analysis conclusion

In the previous paragraph, technical as well as business requirements and recommendations for what-if analytical system were discussed. We explained differences between ROLAP and MOLAP architecture and appointed advantages and disadvantages of particular solutions. For acquiring the best results, recommendations for suitable business implementation were also discussed. The reader should perceive that while properly implemented, it could be strong device in hands of executives. In the following paragraph we will look on areas, where CPM and what-if analysis could be employed and how they could help. We will also discuss reasons for choosing financial and banking sector as exemplary for implementing CPM what-if analytical solution.

3. Performance Management's versatility

The objective of Performance management is to help companies improve and optimize their operations across all aspects of business. It is long forgotten that *Executive Information Systems* are primary dedicated for operational and financial departments. Identification of company divisions, where huge volume of information is produced and where four CPM questions would be able to improve performance is very simple. In the next table, example of five different departments with questions they solve is shown. Biggest attention will be placed on *what-if* question, which interferes in all three pillars of CPM and which will be also key query and object of our interest in case study and whole paper.

Division	Description	Question	Solved problem
Operation	BI has an active role in helping management meet their operational performance measurements. Observed areas could be store or department, specific merchandise, loss prevention, risk, or cash flow. Manager has the ability to monitor performance metrics, analyze information, make proactive decisions,	How?	Is turnover of specific store in defined boundary?
		Why?	Why is operative cash-flow in specific store negative?
		What?	How many of specific goods are planned to sell during 4th Quartile?
		What-if?	How would enlarged selling area influence sales and revenue?

	and act on those decisions.		
Finance	BI provides immediate access to financial budgeting and forecasting data. That enables business decisions to be made based on current and accurate financial data, such as personalized views of revenue information by product, customer, region, branch, and time period. It also enables business decision makers to develop trend revenue forecasts with accuracy and speed, to compare and contrast revenues with the goals, and identify the areas in which the company is performing better or worse.	How?	What is the profitability of particular customer?
		Why?	Why are costs of specific product so high?
		What?	What investments will be made next year?
		What-if?	What will be the profitability of customer, when specific costs will be cut down? How would implementation of ECM solution influence administration costs?
Human Resources	BI supports activities such as recruitment, employee retention, and career development. BI also provides information critical for areas, such as compensation planning, employee benefit planning, productivity planning, and skills rating.	How?	What is current utilization of consultants on projects? When will be additional forces available?
		Why?	
		What?	Who and when will be promoted?
		What-if?	How would additional forces influence our project terms? How would planned promotions influence our revenues and expenditures?
Marketing		How?	How successful were particular campaigns last year? Which was most profitable?
		Why?	Why was this event most expensive? Did it influence our sales?
		What?	How will we use our marketing budget this year? According to last year successfulness, which type of advertisement should we invest in this year?
		What-if?	How would redesign of our web page and increase in page visitors replace current marketing trechniques?

Table 5: Example of solved questions in different divisions

How we can see in the table above, variety of different cases and issues could be solved all across the company. Considering financial crisis or not, one of the most important problem companies solve nowadays is cost minimization and revelation of hidden expenditures. Determination of total costs spent on particular object (product, customer, division, deal etc.) could be very difficult especially in sectors where fixed costs are highly above variable. The main feature of fixed costs is that they are not directly interconnected with final output which produces the revenue. In case of majority, they could significantly warp the total amount of resources spend on particular objects and often manage company to loss. The method which solves this problem is called *cost allocation*.

As we mentioned above, cost allocation is significant especially in sectors where fixed costs are highly above variables. This condition best fulfills financial and banking sphere where variable costs spent on customer or product are minimal and crucial part of expenditures is generated by their fixed component. There are some more reasons for choosing *F&B* sector as the model organizations for further discussion and CPM implementation:

- Huge amount of various data.
- Certainty about their deposit in Data Marts and Data Warehouses.
- Numerous different products, services, customers, employees, branches and divisions on which we can allocate.

- Large number of input dimensions from which we can examine the reality.
- Large number of elements in dimensions, for example millions of customers in clients dimension

After the previous causes, it might seems that only reason for choosing financial and banking sector for applying cost allocation is their massive data basis. It is undoubtedly the primal, however not the only. To the others could be included:

- Interest of the sector for the problematic of cost allocation
- Integration with planning, support for planning decisions
- Possibility of implementing difficult cost allocation models
- Small change could have enormous impact on results and KPIs
- Integration of results to the reporting systems, creation of advanced reports
- Customer oriented approach – change will have also positive impact on customer. For example, cost allocation backend engine could be integrated with front end web interface for products packages modification. Client is then able to easy select products he requires in his package as well as set their parameters. Cost allocation engine will immediately calculate profitability of defined package and offer him monthly price for this combination. This approach will be favorable for bank, which will gain predefined profit from every package as well as for client who undoubtedly satisfy his demands.

After the explanation of reasons for choosing banking and financial sphere as appropriate cost allocation consumers, we will take a closer look to its problematic.

4. Cost allocation

In general, allocation is defined as calculation of cost, margin, revenue, fee or any other quantity to the naturally expressed unit of production (product, labor, service or any operation which is necessary to fulfill). Most often form of the calculation is allocation of costs to the particular external production, i.e. products sold to external customers on the market[12].

In this publication, only allocation of costs as the most important and discussed problematic will be solved. For the better understanding of principles, simple example will be presented in the next paragraph.

4.1. Simple example

Small Software Company develops software. It has 15 programmers and possesses one high-performance server. Closer information about software they develop is in table below.

	Software properties				
Software	Type	Number of concerned programmers	Server utilization (in machine hours)	Total development time (months)	Price
SW A	Enterprise Content Management	5	7 200	12	2 400 000 CZK
SW B	Corporate Performance Management	10	800	12	5 400 000 CZK

[12] Fibrílová J., Šoljaková L., Wagner J.: Nákladové a manažérské účetníctví, page 111

Salary is estimated to 200 CZK per developing hour for ERP programmer and 250 CZK for CPM programmer. They do not have fixed payment, however it depends on programming hours.

Salary properties		
SW Group	Number of developers	Salary per hour
ERP group	5	200 CZK
CPM group	10	250 CZK

The high-performance server they own cost 3 200 000, is linearly depreciated and its estimated life time is 4 years. Therefore one year depreciation is defined for 800 000 CZK.

How could Software Group Manager determine total costs of each software and net profit they produce?

Because salaries of programmers depend on hours they develop software, we can consider them as an variable costs. If they do not code any software, salary costs will be zero. The more hours they develop, the higher salaries are. This is the simplest indicator of variable costs. Manager knows that both softwares are developing whole 12 months. Month has 20 working days, every day 8 working hours. Therefore he can calculate total variable costs for both softwares as follows:

Total variable costs						
SW	months	days	hours	programmers	salary	total
SW A	12	12 x 20 = 240	240 x 8 = 1 920	5 x 1920 = 9 600	9 600 x 200 = 1 920 000	1 920 000
SW B	12	12 x 20 = 240	240 x 8 = 1 920	10 x 1920 = 19 200	19 200 x 250 = 4 800 000	4 800 000

When subtracting variable costs from the price of the software, does Group Manager receive net profit?

price – variable costs	
SW A	2 400 000 CZK – 1 920 000 CZK = 480 000 CZK
SW B	5 400 000 CZK – 4 800 000 CZK = 600 000 CZK

And what they do with the server depreciation? It represents fixed costs? How does Manager know it? Whether they will develop or not, server depreciation will cost 800 000 CZK per year. However, there occurs one problem. How should accountant of the company split 800 000 CZK between two products? Equally? It would not be very accurate and also will warp the actual net profit of products. Manager decided that server depreciation could be divided according the number of developers working on both products, because they all use it.

Number of programmers			
SW	Number of developing programmers	Total developers	Product ratio (number of concerned employees / total developers)
SW A	5	15	5 / 15 = 33,33 %
SW B	10	15	10 / 15 = 66,66 %

The ratio of developers working on both software is 33,33 : 66,66. Therefore also 800 000 CZK will be divided into products in this ratio.

Allocation ratio				
SW	Product ratio	Depreciation (Fixed cost)	Calculation	Allocated cost
SW A	33,33 %	800 000	800 000 * 0,3333 = 266 640	266 640 CZK
SW B	66,66 %	800 000	800 000 * 0,6666 = 533 360	533 360 CZK

After the allocation, Manger is able to determine net profit of both products.

price – variable costs – fixed costs	
SW A	2 400 000 CZK – 1 920 000 CZK – 266 640 CZK = 213 360 CZK
SW B	5 400 000 CZK – 4 800 000 CZK – 533 360 CZK = 66 640 CZK

As we can see, software A (Enterprise Content Management) generates more than three times higher profit then Corporate Performance Management Software B. After this observation, he decided that Company will further focus on Content Management development and unprofitable Performance Management area will not be developed any more. However, after some time, Company is doing much more badly. How it is possible?
If we change the parameter, according what will be depreciation of server allocated to machine hour, the net profit will look like in tables bellow.

Server utilization			
SW	Server utilization (in machine hours)	Total server utilization per year	Server utilization ratio (machines hour for product / total machine hours)
SW A	7 200	8 000	7 200 / 8 000 = 90,00 %
SW B	800	8 000	800 / 8 000 = 10,00 %

The ratio for server utilization is now 90 : 10. Therefore also 800 000 CZK will be divided into products by this ratio.

Allocation ratio				
SW	Server utilization ratio	Depreciation (Fixed cost)	Calculation	Allocated cost
SW A	90,00 %	800 000	800 000 * 0,90 = 720 000 CZK	720 000 CZK
SW B	10,00 %	800 000	800 000 * 0,10 = 80 000 CZK	80 000 CZK

And then we can calculate net profit.

price – variable costs – fixed costs	
SW A	2 400 000 CZK – 1 920 000 CZK – 720 000 CZK = - 240 000 CZK
SW B	5 400 000 CZK – 4 800 000 CZK – 80 000 CZK = 520 000 CZK

If Manager looks at the net profit of both products, he can observe how fatal mistake he made. When wrong cost allocation was executed, management of the company thought, that Performance Management software is less profitable then ECM. However, after the reallocation with another driver, they realized that major costs for server are consumed by ECM solution and that it is highly unprofitable. Although, ECM was three times more profitable after the first allocation according to number of employees, when allocating second time with different driver, it was loss-making and CPM on the other side very profitable.

In the previous example we can experience basic principles of cost allocation as well as observe how important it could be for the company. In the next paragraph, it will be theoretically summed-up and solved in bigger detail.

4.2. Theoretical summing-up

At the previous example, fundamental principles of cost allocation were discussed. However application of this method in real life will be much more difficult and will require greater understanding and expertness of the topic. Therefore, in the further paragraphs we will summarize most important principles and clarify additional possibilities of the method.

4.2.1. Types of costs

4.2.1.1. Variable costs

Variable costs are spent according to the volume of production. They can be directly connected to the product, customer or any other object which is part of the processes in the company. Their total amount is influenced by volume and structure of production[13]. Simple examples are: postage, phone, energy, material, payments etc.

4.2.1.2. Fixed costs

Fixed costs ensure conditions for specific activities in particular extension, therefore are also known as *potential*. They are usually spent in repeating time period. Extension of activity (volume of production) which is possible to create with fixed costs is limited by their capacity[14]. It is difficult or impossible to link them to particular objects in company and therefore they are subjects of allocation. They represent plumbless item in company and can significantly influence net profit of offered products or services. Simple examples are: ICT services, rent, administration, marketing, research and development etc.

4.2.2. Cost Drivers

Cost drivers are usually quantitative indicators, expressly assigned to the objects on which fixed costs should be allocated. In case of existence more drivers according which is possible to allocate costs, driver's package is created[15]. John Shank and Vijay Govindarajan list cost drivers into two categories[16].

- **Structural cost drivers** that are derived from the business strategic choices about its underlying economic structure such as scale and scope of operations, complexity of products, use of technology, etc.
- **Executional cost drivers** that are derived from the execution of the business activities such as capacity utilization, plant layout, work-force involvement, etc.

[13,14] Fibrílová J., Šoljaková L., Wagner J.: Nákladové a manažérské účetnictví, page 143

[15] Popesko B.: Moderní metody řízení nákladů, page 117

[16] Wikipedia: Cost drivers

Example of Driver's package:

	Branch A	Branch B	Total
Number of employees	23	47	70
Total occupied area	900 m^2	300 m^2	1200 m^2
Value of long-term assets	24 000 000 €	76 000 000 €	100 000 000 €

Table 6: Driver's package, absolute values

Selected keys are then recalculated to the percentage ratios, by which are multiplied fixed costs.

	Branch A	Branch B	Total
Number of employees	32,85 %	67,15 %	100 %
Total occupied area	75 %	25 %	100 %
Value of long-term assets	24 %	76 %	100 %

Table 7: Driver's package, relative values (calculated)

4.2.3. Drivers Assignment

In case of multiple keys, user can simply combine drivers and observe impact of selection to the allocation of costs. The more parameters we can identify to the fixed cost, the more options we have while analyzing costs.

Fixed costs	Selected driver
ICT services	Number of employees
Rent	Total occupied area
Administration	Value of long term assets

Table 8: Example of driver's assignment

Easier manipulation and more user friendly interface could be enabled by using drop-down boxes of list of offered drivers. Selected value is then handled as index to the model for further calculations.

4.2.4. Types of cost centers

Any fixed cost created in the company could be assigned to the appropriate cost center. Cost center (division) is a specific part of the organization, working on specific tasks. In general, we can identify two main types of cost centers[17]:

- Production centers
- Support centers

This distribution is very universal and could be used for any of the business activities, not only for the production companies, but also for services, banks, financial institutes, schools, hospitals, etc.

[17] Fibrílová J., Šoljaková L., Wagner J.: Nákladové a manažérské účetníctví, page 48

4.2.4.1. Production centers

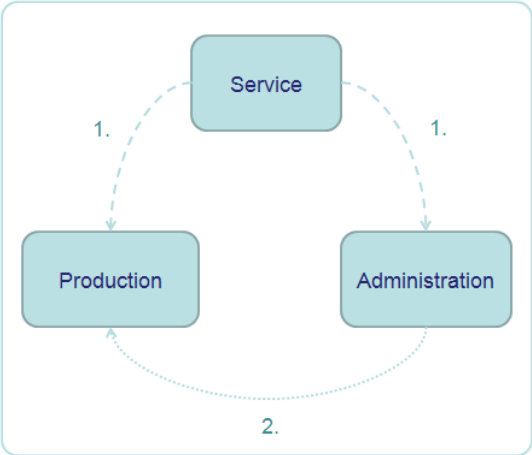
Production centers directly participate in creation of external effort. Goal of these centers is to contribute to the profit of the company. Example of the production centers could be: bank branches, manufactory, legal division, store, etc.

4.2.4.2. Support centers

Support centers include divisions which are not directly producing profit of the company. However their existence is inevitable and they offer services and support to the production centers or creating conditions for primary activity. They could be further divided into support centers such as ICT services or maintenance, administration centers including executive and managers divisions and strategic centers aimed on marketing, research or development. Existence of these centers could be very expensive for small companies and therefore they usually outsource them[18].

4.2.4.3. Reasons for dividing cost centers in cost allocation

Production as well as support centers create different fixed costs during the accounting period. However, support centers do not create any performance; they do not directly generate any profit and therefore have to have zero costs on their accounts at the end of the accounting year. All of their activity (quantitatively represented by costs in accountancy) is focused on supporting production centers. Every production center needs different amount of support (represented by drivers) and therefore we need to allocate these costs to the particular production centers in order to determine how much of support do they received. Fíbrilová, Šoljaková and Wagner describe in their publication flow of costs, as we see in picture bellow, among three types of cost centers – production, service and administration. Firstly costs from service center are allocated to production and administration, in second step; costs from administration are calculated to the particular production centers.



Picture 6: Allocation among different centers, Fíbrilová, Šoljaková, Wagner: Nákladové a manažéské účetníctví, page 132

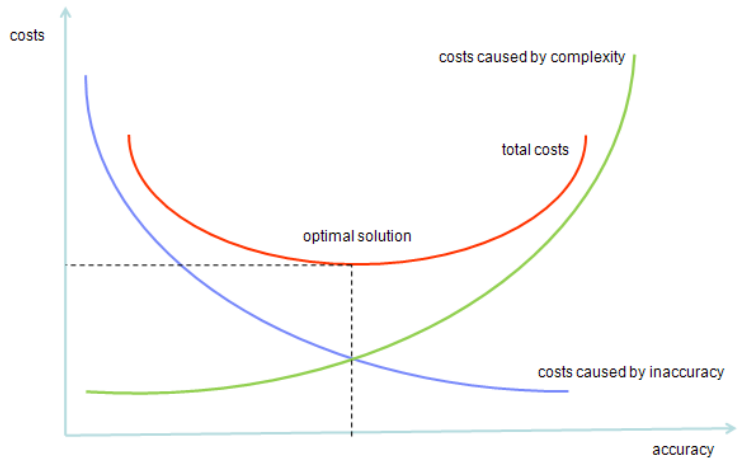
4.2.5. Types of cost allocation

While considering different levels of allocation difficultness, we can identify three main types[19]:

- Simple allocation
- Waterfall (cascade) allocation
- Reciprocal allocation

[18] Fibrílová J., Šoljaková L., Wagner J.: Nákladové a manažérské účetníctví, page 48
[19] Popesko B.: Moderní metody řízení nákladů, page 138

Each of allocation varies not only in difficulty level on one side but quality of output and similarity to real processes and cash-flow on the other. Selection and implementation of method will depend on user criteria. He has to consider many factors, such as size of the company, number of objects (centers, products etc.), accuracy of output, time as well as resources invested to the model development, available data sources, number of drivers, etc. Particular methods will be explained and described further in the text. Kaplan and Cooper visualize total costs spent on implementation, which enable selection of optimal method as conjunction of two lines. While implementing difficult and real life model, increasing accuracy lowers costs caused by inaccuracy. On the other hand, costs caused by complexity of the model are rising. As we see, optimal solution will offer maximal accuracy with minimal invested resources[20].



Picture 7: Optimal solution of cost allocation model, Kaplan, Cooper

4.2.5.1. Simple allocation

In this method, all business divisions could allocate data only on the lowest level of other dimension. It means that in case of allocation on product, all departments (units) have to set up (load) drivers that allocate costs directly to product level. Total cost for defined units is simply split between products. This allocation could be done in one step. Calculation is based on drivers. Example of matrix for drivers is visible in table below[21]:

		PRODUCTS					
		1	2	3	4	5	Total
UNITS	A	30 %	10 %	20 %	30 %	10 %	100 %
	B	10 %	25 %	35 %	10 %	15 %	100 %
	C	20 %	30 %	20 %	10 %	20 %	100 %
	D	0 %	50 %	0 %	50 %	0 %	100 %

[20] Popesko B.: Moderní metody řízení nákladů, page 121
[21] Bothe O., Pavlík M.: Types of cost allocation

4.2.5.2. Waterfall (cascade) allocation

In this case an order of business unit is set up. The first unit could allocate costs on all other units and on detail level (for example product) as well. The second unit could allocate data directly on the lowest level of product and also on all units except the first one. The last business unit could allocate data only on the detail level.
The calculation is done in several steps. In the first step, the first unit allocates cost on other unit and products. In the second step, the second unit adds costs allocated from the first unit to its costs. This total is allocated on other units and products. Last units add their costs to costs from all previous units and allocate this total only on product level. Calculation is based on drivers. Example of matrix for drivers is visible in table below[22]:

		Units				Products					
		A	B	C	D	1	2	3	4	5	Total
Units	A	x	10 %	20 %	30 %	10 %	10 %	10 %	10 %	0 %	100 %
	B	x	x	50 %	0 %	10 %	10 %	10 %	10 %	10 %	100 %
	C	x	x	x	40 %	20 %	30 %	0 %	10 %	0 %	100 %
	D	x	x	x	x	0 %	0 %	50 %	50 %	0 %	100 %

4.2.5.3. Reciprocal allocation

This allocation method is the most difficult one. All units could allocate data on detail level or on all other units. This allocation runs in several rounds, each round has got several steps. Each round includes allocation between all units and all products and units. After this process, costs allocated on units are summarized (by units) and used as input for second round of allocation (instead of total costs that was used in first round). Settings of drivers have to be done properly. There are cases when this allocation method does not lead to desired outputs – cost on units is not decreasing. In these cases it is possible to check if the current setting of drivers leads to cycle or not. Calculation is based on drivers. Example of matrix for drivers is visible in table below[23]:

		Units				Products					
		A	B	C	D	1	2	3	4	5	Total
Units	A	10 %	10 %	20 %	20 %	10 %	10 %	10 %	10 %	0 %	100 %
	B	0 %	0 %	50 %	0 %	10 %	10 %	10 %	10 %	10 %	100 %
	C	10 %	10 %	0 %	20 %	20 %	30 %	0 %	10 %	0 %	100 %
	D	0 %	20 %	30 %	0 %	0 %	0 %	0 %	50 %	0 %	100 %

4.2.6. Conclusion of theoretical summing-up

After familiarizing with main stones of cost allocation, we will summarize steps which will be used in following case study and which we described in previous paragraphs.

1. We need to transform relational data from primary systems (or DW) to the binary OLAP analytical application in order to execute *what-if* analysis.

[22] Bothe O., Pavlík M.: Types of cost allocation
[23] Bothe O., Pavlík M.: Types of cost allocation

2. We need to build define what are variable cost, what are fixed costs and on what detail we want to allocate.
3. We need to set-up or load drivers.
4. We need to assigned drivers to the particular costs.
5. We need to select optimal method for allocation in order to mineralize costs and gain maximal accuracy.
6. We need to implement this solution.
7. We need to execute what-if analysis.

These steps will be described in detail in further part of the paper called *Case Study*.

III. Case study

5. Introduction to the case study

In the first part of the paper we explained reasons and necessity of implementing CPM solution with what-if analytical capabilities as well as we declared arguments for exploiting cost allocation model in banking and financial sector. In this part of the article, we will simulate layout of the project and its implementation focused on reciprocal cost allocation method in mentioned sphere. Gradually, all common parts of the regular IT project will be solved and explained.

We will start with the company's order declaration and specification of requirements. Then analyze of the recent data sources will be done and structure of the objects in the DW will be explored. After the examination of the source systems, design of the future application and its logic will be specified. This part will be followed with the self implementation, processes development and UI proposition. At the end, two exemplar what-if analyses will be executed.

6. Order specifications and requirements

Despite the grooving market share and number of customers, executives of the fictive bank are not satisfied with its profitability and net profit it produces. Financial accountancy revealed that high incomes have been compensated with grooving rate of expenditures, especially of fixed character. Executives of the bank blame key divisions for their high spending and require retrenchment. However every division manager claim that his center directly contributes to the performance of the company and its costs can not be cut down. Because of the high number of different costs and centers, managerial accountancy they conduce is insufficient and is not able to identify objective contribution of particular divisions. This is the first reason why board decided to implement application for cost allocation calculation.

To simulate the flow of money among the company as real as possible and to trace torrent of costs in particular steps they decided to implement the most authentic - reciprocal method.

After the advantages of Performance Management where explained to them, it was decided to implement it in MOLAP technology as the new layer above the Data Warehouse. From contribution it brings will be for managers also interesting:

- Ability to choose from more cost drivers and observe the changes in money flow among the divisions and final impact of allocated costs to particular products
- Along with the cost allocation cubes, ability to store and observe another data (sales, variable costs, prices) in multidimensional environment
- Their cohesion and ability to calculate profitability of products
- Ability to execute what-if analyses over whole model
- Easy integration with reporting tools

For implementing this CPM what-if analytical solution I selected IBM Cognos TM1 software, which meets all of declared requirements. In the next paragraph, IBM Cognos TM1 and its specifications will be introduced closer.

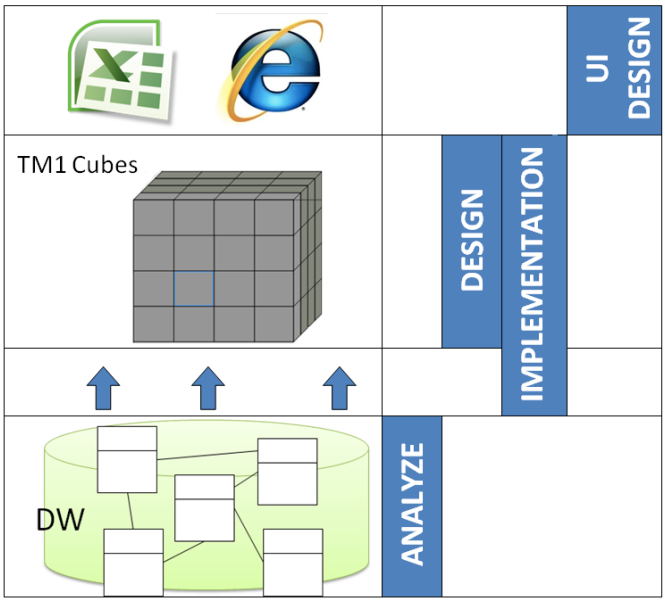
7. Selected technology

Project will be implemented in IBM Cognos TM1. TM1 is multidimensional, 64bit, in-memory OLAP analytical engine, which enable write-back capabilities. Therefore is suitable for creating and working with what-if analytical applications. Thanks to the patented engine, it achieves high performance while processing large amount of data or complicated calculations. Placed in the middle of the BI architecture, it is necessary to offer easy and compatible import and export to wide range of data sources. TM1 includes special data processor called Turbo Integrator, which enables easy connection and load from many data sources using ODBC and ODBO connector, or accessing ASCII files and SAP. On the other hand, processed data from TM1 could be exported for presentation and further processing to MS Excel, published on the TM1 web interface or integrated with professional reporting devices.

Starting with the next paragraph, we will describe architecture of the solution as well as particular steps of the project. In these parts, some of the mentioned approaches to the import and export will be explained in more detail.

8. Project solution

As we can see in the picture bellow, TM1 analytical engine is placed between primary data sources (represented by Data Warehouse) and presentation layer, which is in this case MS Excel published on TM1 Web interface. In the rectangles on the right side of the architecture, particular steps of the solution as well as area they affect are shown. Firstly, structure and data provided by Warehouse will be analyzed. After that, we will design appropriate architecture of TM1 cubes, with regard to functionality and results customer requires. This will be followed with implementation of designed cubes as well as processes and reciprocal cost allocation development. At the end, user interface in MS Excel will be prepared and published on the TM1 Web. In the next paragraph, first step of the solution – data source analysis, will be discussed.



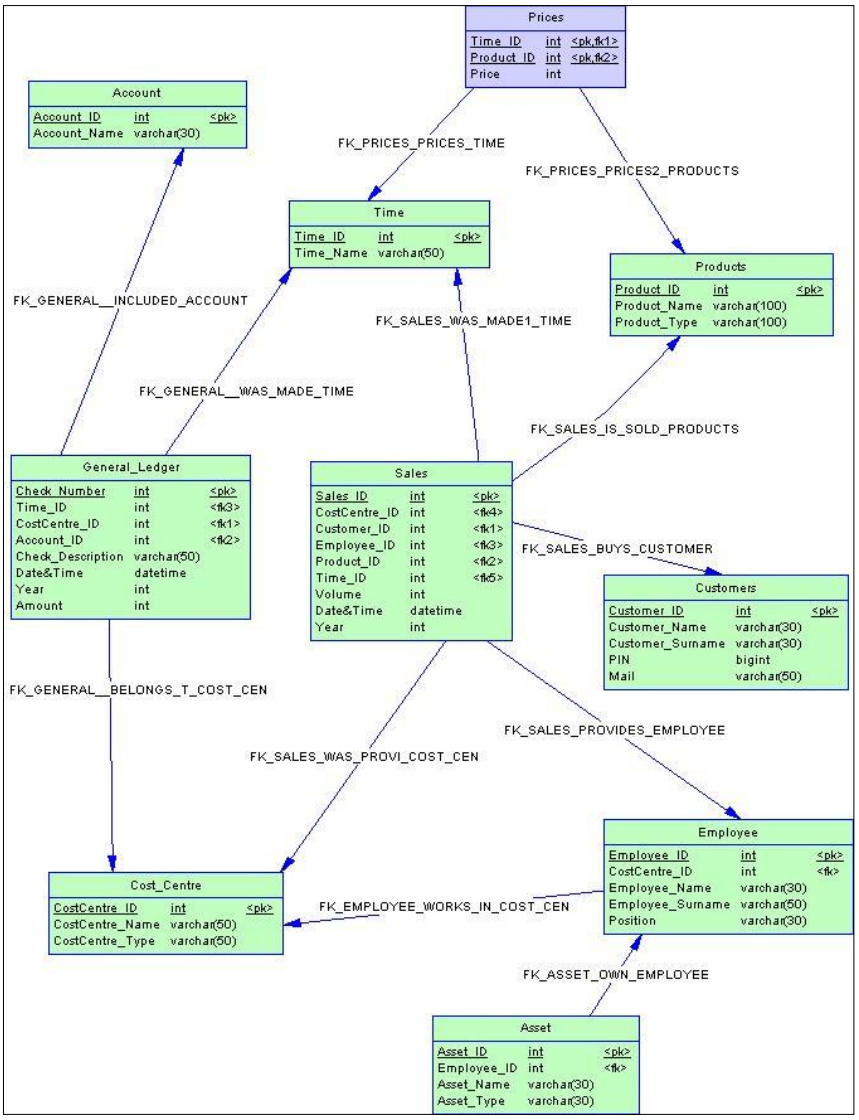
Picture 8: Particular steps of project solution

8.1. Analyze of input data sources

As was mentioned in the paragraph describing differences between ROLAP and MOLAP technology and basic requirements for what-if analytical system, OLAP layers are primary designed to support managers and executive decisions. It means that they do not produce any data (they appear as data consumers and processors of existing entries) and therefore need to be loaded from the primary systems. Loading from the primary sources and transformation to the multidimensional (binary files) requires awareness about the structure of object (relational tables) and data they store. It is why the solution of future CPM cost allocation application will start with data sources analysis.

As the initial data source for reciprocal cost allocation model we will use very simplify version of bank's central Data Warehouse. In the following picture, graphical representation of DW physical model is shown.

Now, we will not concern with the individual objects in the model. It will be very prosy and some tables will even not be needed. However, we will many times reference and backspace to its structure while designing and implementing particular cubes in OLAP layer. Then we will in detail observe particular tables and manipulate them to achieve sources for loading into the cubes.



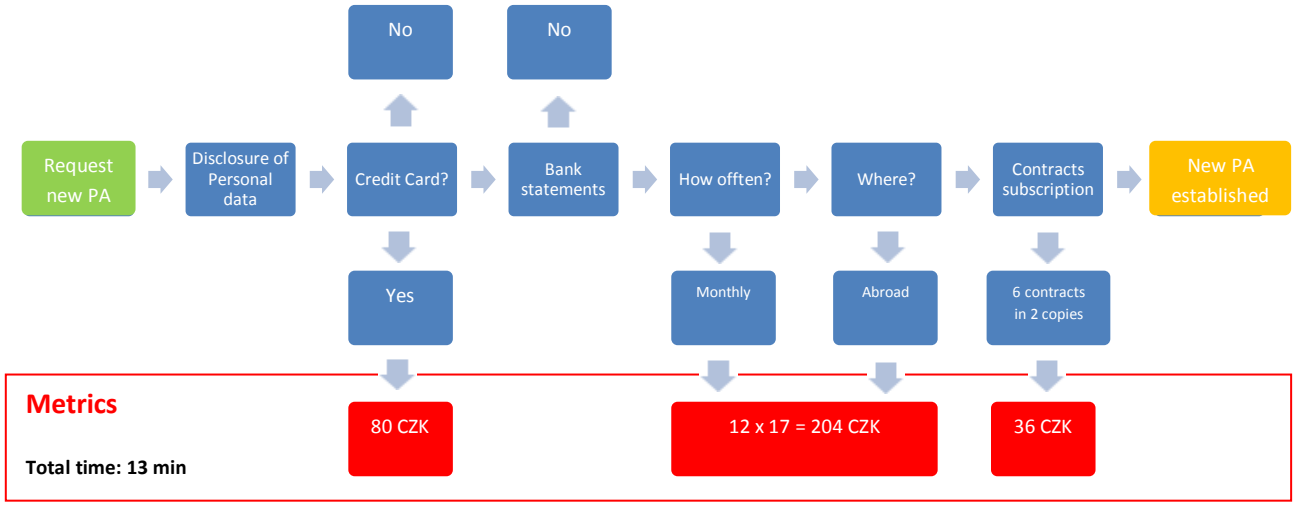
Picture 9: Physical model of Bank's Data Warehouse

However, not all of required input entries are stored in Data Warehouse. The complexity and access to all of them is considered to be the one of the most important problem while building CPM application. Many times data are not stored in databases or are very difficult to access and collect. Therefore we need to specify all available channels and approaches how we can retrieve them. One most common in my opinion will be used also while implementing cost allocation. It is:

8.1.1. Outputs of processes

Many repeating activities are nowadays conducted according to foreordain processes. Process describes particular operation as a flow of continuous actions and events, which lead to the required result. Every activity can produce outputs and metrics as well as consume resources. These will be in interest of CPM systems, because are able to represent and provide important data for performance monitoring, optimization and managers decisions. For better understanding, simple example with metrics will be shown:

Establishment of new personal account:



Picture 10: Process of new personal account establishment

As shown in the picture above, pre set metrics of particular *products establishment processes* will represent special group of data inevitable for accurate CPM solution while offering many possibilities for what-if analyses optimization. In the exemplary case study of cost allocation, we will use this data for variable costs collection and total direct costs calculation. Because of their relatively firm and not often changed value, they will be inserted into the model by hand.

However, outputs from the processes do not have to be only financial character. Another group of valuable metrics will represent for example time, utilization, steps, consumption of sources etc. In exemplary process above, valuable information for cost allocation will be average time teller spent with client while processing the contract. This metric will be appropriate for cost driver which allocate salaries of bank employees in branch among particular products. Driver could be for example:

Number of established account per mount x time spent on processing the contract

This value provides information about total time spent on product in one month. Divided by total time of employees in bank branch represents percentage of their salaries redound to the considered product.

8.1.2. Other sources

Another possibility how to collect valuable data for accurate Performance Management solution could be outputs from different marketing researches, help desk reports, service center statistics or measurements from other systems such as hardware and infrastructural performance, what is very relevant to banking sector. This methods are however very advanced and probable to implement only in large solutions. In the reciprocal cost allocation case study will not he used.

8.2. Architecture and design of the cubes

After the analyze of input data sources as well as way how we can retrieve the necessary one, project can be moved to the next step, which is architecture and design of the cubes. Fruitfulness of the whole solution as well as its flexibility, further extension and proper connection to the input data sources can be significantly influenced by this step. Therefore, every project manager should access to this topic very preciously and with the great knowledge of required results as well as capabilities of using technology. Wrong design of cubes could bring required results, however its flexibility to the changed or additional demands and adaptability to the varied data source could be impossible.

Because we are well awarded with that treat, approach we selected and observe is division of projection to the particular steps. During the each step we will be piloted by main demands and requirements which are retrieved from client through right formulated questions. It enables us to design flexible and proper structure as well as particular cubes and dimension they include. This approach ensures that every demand will be considered and included in the solution. It also enables preparation to the further extension or presumably occurred changes.

Question 1

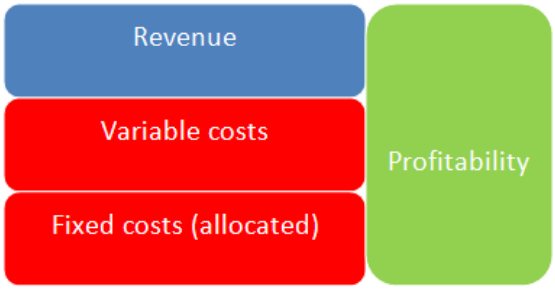
Aim: To intercept concept of the solution, its gross structure, objectives we will try to achieve.

Question: *What results do you expect from the cost allocation application?*
Shortened answer: *reciprocal cost allocation calculation with profitability of products observation*

Solution: From the answer we find out, that the goal of the application is to observe impact of allocated costs to the products profitability. Profitability is in general calculated as:

Profitability = revenue – variable – fixed (allocated costs)

This information will significantly influence the structure of the architecture. For the independent processing, calculating and observing of individual entities of profitability, we will primary divide concept to four main parts. Each part will serve for singular component of profitability and enables its possible modification without impact to the rest of the solution. The structure is displayed bellow.



Picture 11: Main concept of the solution

Every component will handle mutual or different cubes to provide results for final consolidation in profitability cube. In the next question we will try to closely design singular parts, ways how they receive data from primary source and provide calculated one to the result cube.

Question 2

While we cogitate 4 single parts of the solution, question 2 will be divided into 4 subparts providing information to each component. In every one, we will try to ordain proper cubes. Required dimensions in these cubes will come up from clients demand for detail he wants to observe in result (profitability) cube. The reverse approach will be therefore selected. Firstly we have to design final cube with required dimensions and then trace back to particular components and design their dimensions. Therefore the initial question will be:

Question: *On what detail do you want to observe products profitability? Do you require any special dimensions which are not available so far in the Data Warehouse?*
Shortened answer: *Profitability of individual products, products types and all products according to singular quartiles. Total profitability of particular products and profitability per unit. Do not want to store historical data or any other detail, however requirement for building and modeling different scenarios and possibility to compare them.*

Starting points for subparts: The answer provides**,** that we need to store and handle data in every component about whole product dimension on every level of hierarchy. *Revenue* part, *variable costs* part as well as *fixed costs allocation* therefore have to include *products* dimension. Time dimension will handle only quartile elements and year dimension is unnecessary considering the fact that client do not want to store historical data. They just want to load current data, execute allocation and observe results. These need to be compared and therefore *version* dimension with particular scenarios is inevitable. Measure dimension called profitability metrics will include for every product total revenue, total variable costs, total allocated costs, total net profit calculated as shown in the equation in question 1 and profitability per unit estimated as total net profit divided by number of sold products. These are all information we retrieve from the initial question 2 and enable us to design result cube. It can be summarize in following table.

Profitability cube

Profitability cube			
Dimension name	**Hierarchy**	**Elements**	**Note**
Products	0 Products 1 Products types 2 Total Products	Loaded from DW	Could be changed
Time	0 Quartile 1 Total Year	Q1, Q2, Q3, Q4	
Version	none	Original	Could be added and modified
Profitability metrics	none	Revenue Variable costs Fixed (allocated costs) Net profit Net profit / unit	 = Revenue – VC - FC = Net profit / sales count

Having the required result cube, we can continue with back tracking the source cubes for the profitability one in another three components. Subpart of question 2 will start with the simplest one – Revenue calculation. To calculate the total revenue, we need to find out how client calculates it.

Revenue component

Aim: To find out how to calculate revenue component.

Question: *How do you calculate revenue? What influences it?*
Shortened answer: *Revenue is calculated according to product type. We offer two types of products – deposit and credit accounts. Total Revenue for deposit accounts is calculated as number of sold accounts multiplied by their price and credit accounts as volume of provided credit (loan, mortgage etc.) multiplied by interest rate which is current for this type of product.*

Solution: The answer provides that total revenue is calculated as multiplication of two values. Therefore, revenue component will be divided into two singular cubes handling sales count and volume in one and prices and interests in the other. Now we have to determine dimensions in every cube and estimate data source for their filling. Considering the detail of profitability cube, *time*, *product* and *version* dimension have to be at least included in every cube.

Price cube

Price cube			
Dimension name	**Hierarchy**	**Elements**	**Note**
Products	0 Products 1 Products types 2 Total Products	Loaded from DW (could be changed)	
Time	0 Quartile 1 Total Year	Q1, Q2, Q3, Q4	
Version	none	Original	Could be added and modified

Price cube will not require additional dimensions and stored data will be easily accessible from the *prices* table in the Data Warehouse. As we can see from physical data model, this table provides every necessary entry. Credit account will have their interest rates stored in *price* column in percentage expose. Transformation and data load from DW to cubes will

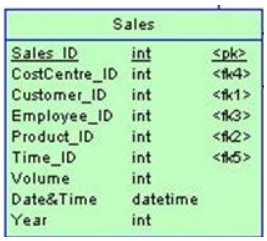
Picture 12: Detail about *Prices* table

be closely described in implementation. For architecture and design phase is inherent only that data exist, are available and accessible.

Sales&Production cube

Sales count cube called *Sales&Production* will include all necessary dimensions, however we will need for every product store two different values (count and for credit products also volume). Therefore another dimension called *sales metrics* will be created. Data will be loaded from *sales* table which provides also all necessary information. Number of sold products in *count* element of *sales metrics* dimension will be retrieved by using *count* query to the table and *volume* of provided credit will use *sum* query for *volume* column.

Sales&Production cube			
Dimension name	**Hierarchy**	**Elements**	**Note**
Products	0 Products 1 Products types 2 Total Products	Loaded from DW (could be changed)	
Time	0 Quartile 1 Total Year	Q1, Q2, Q3, Q4	
Version	none	Original	Could be added and modified
Sales metrics	none	count, volume	Volume only for credit accounts

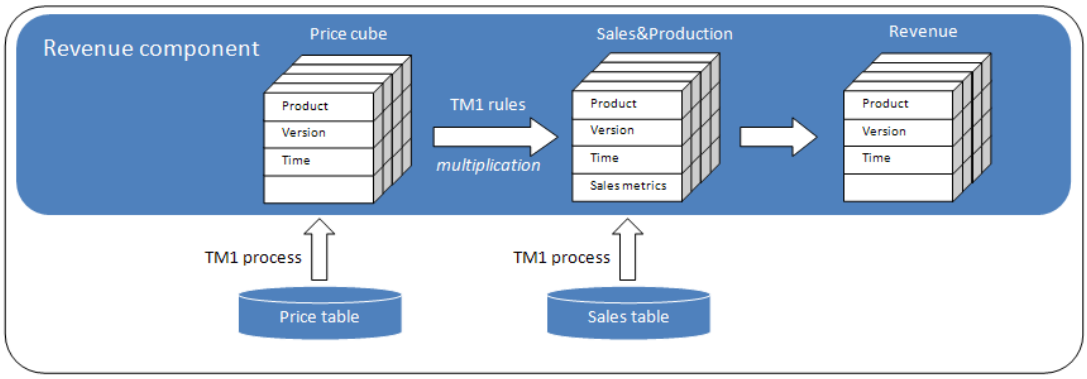


Picture 13: Detail about Sales table

As we can see in the physical model, sales table holds more information than we planned to use so far. For the revenue component we are creating now it is sufficient, however it is not secreted that any other column and dimension will be needy in further parts. In architecture design step of solution it is possible and no complications will occur.

Revenue cube

Data from *Price* and *Sales&Production* cube will be multiplied with one another using calculation rules (will be explained in implementation phase) and will fill the *revenue* cube. Revenue cube include all three necessary dimensions. The whole revenue component of the solution is displayed below and does not need any further explanation.



Picture 14: Revenue component detail

Variable costs component

Having the first component designed, we will move to the second simplest one – architecture of variable costs cubes. To create proper architecture of the solution, we have to get some answers from the client.

Aim: To find out how they calculate variable costs.

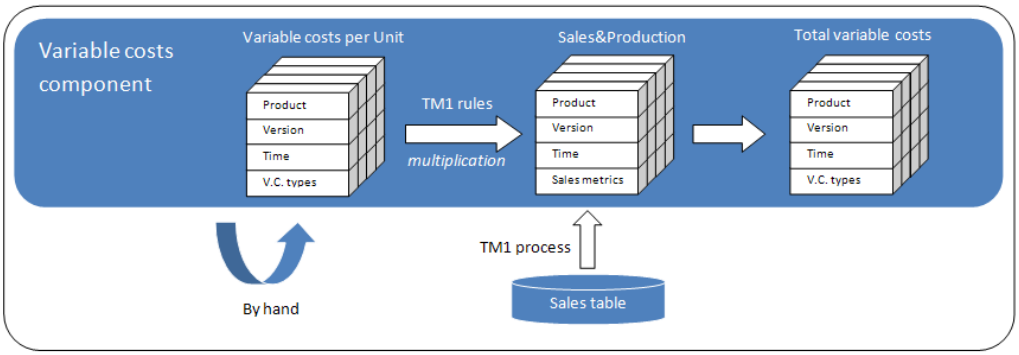
Question: *How do you measure and evaluate variable costs per unit?*
Shortened answer: *For every sales contracting, we have designed special purchasable processes. They measure, evaluate and assess variable costs which are directly connected to the deal. However, they are not yet stored in digital version, only controlling department monitors them for internal demands.*

Solution: Much like in revenue component, we will need to calculate total variable costs for particular products. It will be determined as number of singular products sales (yet designed) multiplied by variable costs per unit. However they are not stored in digital version, cube need to be filled by hand or particular database have to be create for them. Cube will include three necessary detail dimensions as well as measure dimension which elements represent individual items of costs. It will be called *variable costs types* and include members such as phone, postal, paperwork, direct marketing, cards and for credit accounts also interest rate which they pay depositors for liquidity.

Variable costs per Unit cube

Variable costs per Unit cube			
Dimension name	**Hierarchy**	**Elements**	**Note**
Products	0 Products 1 Products types 2 Total Products	Loaded from DW (could be changed)	
Time	0 Quartile 1 Total Year	Q1, Q2, Q3, Q4	
Version	none	Original	Could be added and modified
Variable costs types	none	Phone, post, paperwork etc.	

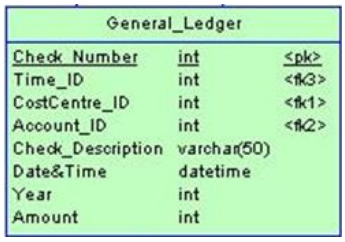
While deciding about implementation of total variable costs, we can choose from two solutions. It could be done as another dimension in variable cost cube or as solitary cube. For higher lucidity, we will prefer second solution and implement *total variable costs* cube. It will include same dimensions as *variable costs per unit* and data will be filled from multiplications of *sales* cube and *variable costs per unit*. In the depiction bellow we can observe final architecture of the component. Sales cube will be mutual with the one in revenue part.



Picture 15: Variable costs component detail

Fixed costs allocation component

Last and the most difficult one will be the fixed costs allocation component. This part will require more questions, in order to meet all customers demand and flexibility of solution. Firstly let's examine structure of fixed costs, we want to allocate, in Data Warehouse. These are stored in table called *General Ledger*. As we can see in the physical data model of the DW, every cost originates in one of the *cost centers* (column 3), is assigned to one of the *accounts* (column 4), belongs to the particular *quartile* (column 2) and has its value (column 8). If we add to this version dimension, input cube for the allocation could be completed. Now, it will be appropriate to observe differences between the detail we have in input cube and results we require from allocation calculation component. Necessary dimensions *time* and *version* are in both cubes. In input cube accrues *account* dimension but we are not able to identify there *products* dimension. Instead of this, fixed costs are captured in *cost centers* detail. Therefore we need to create allocation engine in this component, which transform costs from cost centers detail to product detail. How it could be implemented? Specification of the order requires reciprocal type of allocation. As was explained in theoretical part of the thesis, in reciprocal method every cost center could allocate to any other one and also to arbitrary product. This is depicted in the picture bellow.

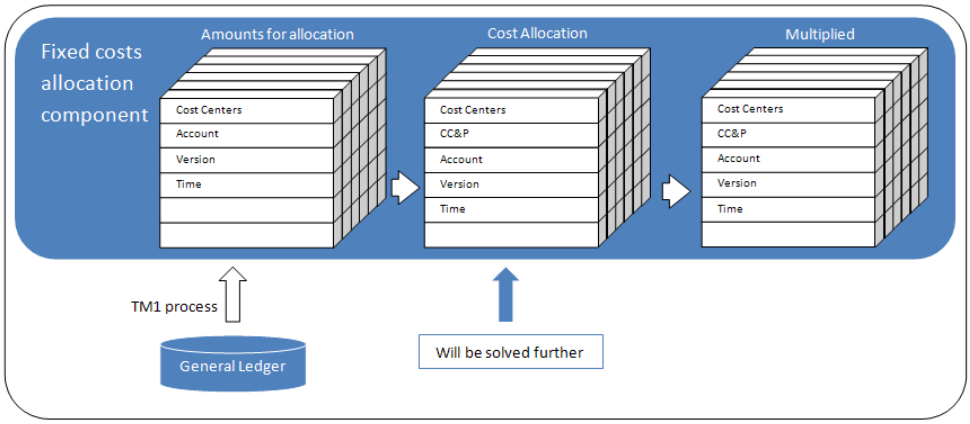


General_Ledger		
Check_Number	int	<pk>
Time_ID	int	<fk3>
CostCentre_ID	int	<fk1>
Account_ID	int	<fk2>
Check_Description	varchar(50)	
Date&Time	datetime	
Year	int	
Amount	int	

Picture 16: Detail about General Ledger table

	Cost centers				Products			
Cost centers	20 %	30 %	10 %	40 %				
		50 %	50 %					
	10 %				30 %	30 %	30 %	
							50 %	50 %

This cube will handle percentage, which determine how singular cost center contributes to any other division or product. First two dimensions therefore have to be *Cost Centers* and special dimension which is developed as combination of *Cost Centers* and *Products* dimension. We will call it *CC&P* dimension. However, costs in input cube for allocation are stored also in *account* and *time* detail. Therefore, we need to include these two dimensions in order to enable severance of cost allocation percentage on different accounts or in different times. Now, let's review, what cubes and dimensions we have so far. It will greatly help us to comprehend the current model and also will be helpful in further development.



Picture 17: Fixed costs allocation component detail

Reciprocal cost allocation will run in loops until all (or defined percentage) of costs will be allocated to particular products. Every step (one run of the loop) will dislocate costs and move predefined percentage of them to other position. We need to capture this movement and before another step of loop adapt these changes to the input data. In the next picture we will simulate sequence of the loop with currently available cubes. *Multiplied* cube will serve as temporary storage place for allocated values (fixed costs in amounts for allocation x percentage in cost allocation), however we will exploit it further. We also anticipate that 3rd and 4th dimension of every cube as set to *1th Quartile* element in *Time* dimension and *Software* in *Account*.

Amounts for allocation

	Amounts
Cost centers	250 000
	100 000
	300 000
	600 000

TM1 rules multiplication

Cost Allocation

	Cost centers				Products				Total
Cost centers		50 %	10 %	40 %					100 %
	50 %		50 %						100 %
	10 %				30 %	30 %	30 %		100 %
							50 %	50 %	100 %

Multiplied

	Cost centers				Products				Total
Cost centers		125 000	25 000	100 000					250 000
	50 000		50 000						100 000
	30 000				90 000	90 000	90 000		300 000
							300 00	300 000	600 000
Total	80 000	125 000	75 000	100 000	90 000	90 000	390 000	300 000	1 250 000

In multiplied cube we can find allocated values to particular centers and products in first step. Red cells represent amounts which were allocated from the cost centers to other units. For example 1th cost center participate to 2nd cost center with amount of 125 000 etc. These values have to be subtracted from *Amounts for allocation* cube, however on the other hand, if we observe green cells, they represents values, which were allocated back to cost centers. Therefore they need to be added to Amounts for allocation. Before the second step of the loop, situation and values in the cubes will look as follows.

	Amounts
Cost centers	80 000
	125 000
	75 000
	100 000

Now we have to solve problem, whether rewrite data in Amounts for allocation cells or create additional dimension which will store values from particular steps of allocation. We can as client if this information will be needful and important for him or not.

Question: *Do you require seeing how costs were dislocated in particular steps of allocation?*
Shortened answer: *Yes, this information is important for us.*

Solution: We add *steps* dimension to the amounts for allocation cube. This ensures that every step will be noticed in new cells under new element. The slice of cube will then looks like shown bellow.

	Amounts	
	Step 1	Step 2
Cost centers	250 000	80 000
	100 000	125 000
	300 000	75 000
	600 000	100 000

Same problem will occur in *Multiplied* cube. In following step we need to calculate newly allocated values, which could rewrite amounts from previous step or could be add to new element in *steps* dimension. This information will be also very interesting and valuable for clients so we add another dimension into the cube.

Step 2									
	Cost centers				Products				Total
Cost centers		40 000	8 000	32 000					80 000
	62 500		62 500						125 000
	7 500				22 500	22 500	22 500		75 000
							50 000	50 000	100 000
Total	70 000	40 000	70 500	32 000	22 500	22 500	72 500	50 000	380 000

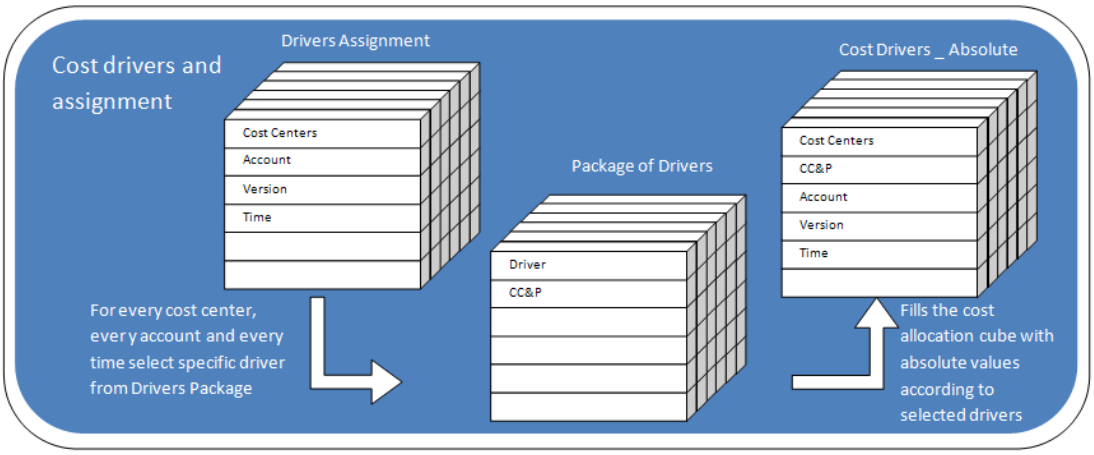
Now, only two unsolved questions remain. Firstly, we have yet not determined (picture no. 17), how will be drivers loaded and assigned to singular cost centers in *Cost Allocation* cube. We just predicted that they are in percentage expose. We try to find out clients imagining and demands.

Aim: To design appropriate concept of cost drivers and their assignment.

Question: *On what detail and how do you require determining and assigning drivers to the costs?*
Shortened answer: *To every cost center, on any account in any time should be possible to assign different cost drivers. Only it ensures maximal accuracy and flexibility in cost allocation. Analyst has to*

be able to changed them and manage (add, remove). Also the user friendly approach to driver's assignment is required.

Solution: According to the requirements, the best solution seems to be as depicted bellow. In the middle cube, analyst will have possibility to manage and change different cost drivers. Every driver has to include amounts, which enable proportional allocation among cost centers and products. Therefore the cube includes two dimensions – driver and CC&P. Through the drivers assignment cube, he will be able to select specific drivers from individual fixed costs. According to selected driver, his values will be sent into the *cost allocation _ absolute* cube. However, earlier we manipulate with percentage drivers. This will be ensured by recalculation of absolute values from *cost allocation _ absolute* to percentage in *cost allocation* cube.



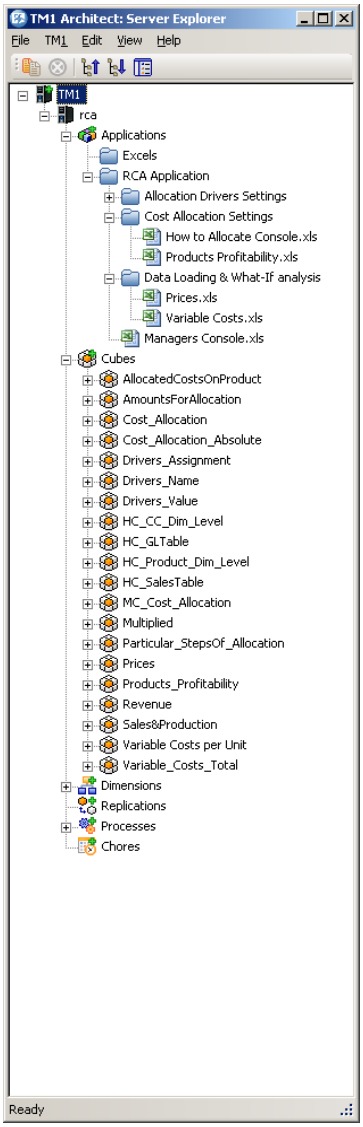
Picture 18: Cost drivers and assignment component detail

The last problem is closely joined with reciprocal method. As we know, it runs in loops, until all costs from cost centers are not allocated into the particular products. However, it brings some difficulties. Drivers are highly possible to be set in circle, which means that cost centers will invert costs with one another, without allocating to products. It could cause circular reference of the allocation process, which will force engine to be restarted. We suggest client to avoid this by using conditions in process. These could be solved in two ways. User can set fixed number of iteration, after which process stops and corresponding part of costs will be allocated. While using this approach, we do not know beforehand, how much of costs will be allocated. In the second method, we set percentage of costs which have to be allocated before process stops. This ensures that required amount will be allocated, however we do not know, how many steps it needs and how long it will last. Because client is doubtful about which method is more appropriate and effective for him, we decided to implement both. The way how we are able to obtain it will be closely solved in implementation part of the project.

Now, we have designed all components of the solution. If we are sure that all needs and requirements are covered, project could be moved to the next step – implementation.

8.3. Implementation

This part of the project covers development of designed model in IBM Cognos TM1 software. After the introduction of TM1 environment, all four components of the solution will be gradually implemented with explanation of used functionality and options. Extension and difficultness of the solved problematic however do not enable me to describe every step of implementation. Although I chose to outline in detail only part which are key and most interesting for development, but more advanced to be describe in available user's documentation. With good architecture and design of the solution, implementation is sometimes very recrudescent and many steps are reiterated. Considering the coverage of the thesis I explain and with help of screen shots depict every operation only once, following similar moves will be just referenced to the already explained parts.



Picture 19: TM1 Server Architect with running RCA application

8.3.1. Introduction of TM1 environment

Referencing to the *Basic requirements for what-if analytical system* paragraph (p. 15), I mentioned advantages of operating with all data in RAM memory. TM1 meets this demand while storing all objects in file system structure and after starting the corresponding server, loading them into the memory. Picture no. 19 shows running RCA (reciprocal cost allocation) server with all components stored on the server. Six main types of them are:

Applications – enable disposal of objects, both internal (TM1 cubes, processes etc.) as well as external (Excel sheets, Word documents) into the transparent structures

Cubes – include dimensions and store data, enable creation of different predefined views and rules for static calculations and data manipulations

Dimensions – set of entities (elements) with one or more common characteristics stored in hierarchical structures, enable creation of static predefined as well as dynamic subsets

Replications – provide data and object synchronization between servers

Processes – enable connection and loading from different data sources and advanced data transformation

Chores – provide possibility to run process (package of processes) in predefined times or intervals

With combination of only these objects we can build a powerful application such as reciprocal cost allocation engine. In contrast to architecture and design part of the solution, where we started with profitability component and reversely proposed another ones, implementation part will be handled from particular components and finally join into profitability results. As the starting point of the

implementation I chose *Revenue* component which is also the simplest one to implement and on which I could explain basic principles of TM1 *processes* and *rules*.

8.3.2. Revenue component

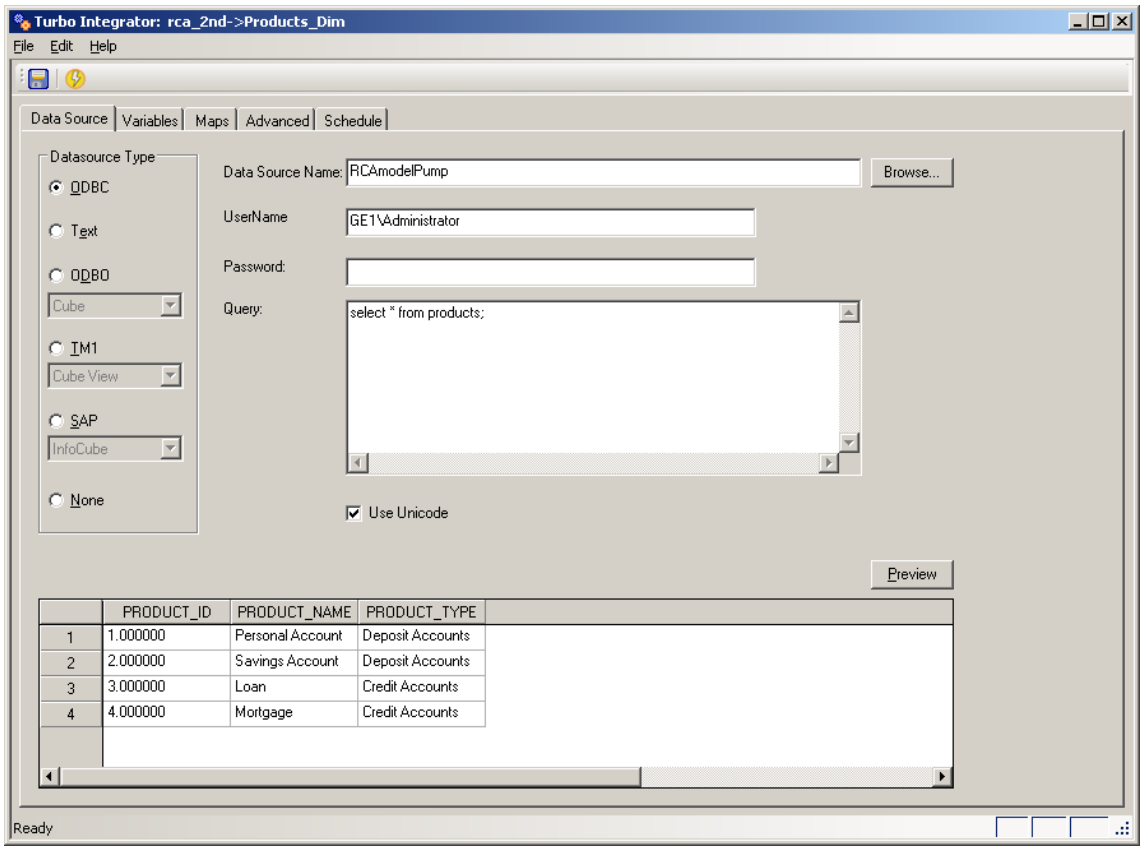
Revenue component is depicted at picture no. 14. Before creating cubes we have to build up dimensions which are included in them. As we can notice from the scheme, all three cubes contain *Product*, *Time* and *Version* dimension. Creating them, we will demonstrate two main ways how could be dimension constructed in TM1.

8.3.2.1. Product dimension

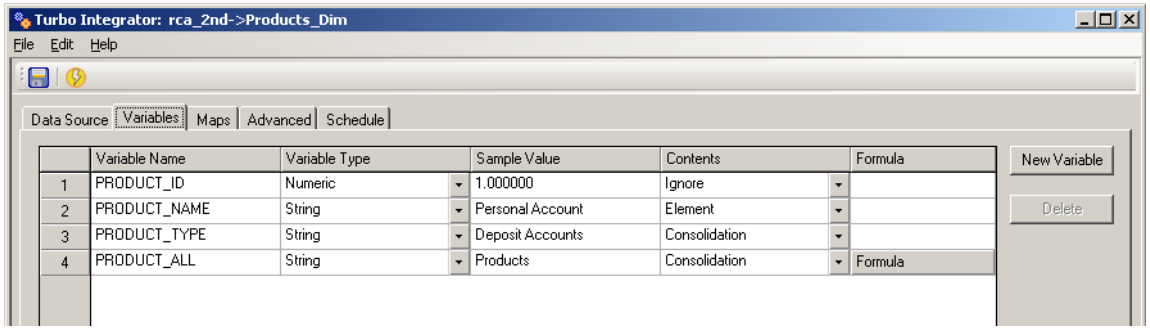
Elements in product dimension are likely to be changed because of the current bank offering as well as dimension itself is being consider to contain numerous entities. Therefore I decided to create it using *TM1 process*. Instead of adding elements to dimension by hand, processes are convenient especially in situations, when we want often and easily reload actual elements or when dimensions contains many entities and their adding by hand could be longwinded. Processes enable connection to the different data sources and creation of dimension from loaded values. During defining the process (in TM1 also called Turbo Integrator) user is able to set and modify many parameters which determine how will be loaded values organized in the dimension. In the next set of pictures, I will show and describe basic parameters defined in TM1 process designed to load Product dimension called *Products_Dim*. Firstly we have to determine data source of Warehouse where table with current offering of products is stored. After creating connector to MS SQL Server 2005 where bank's Data Warehouse is running, we could create SQL query to retrieve required values. As we can see in the picture no. 21, in Data Source tab of Turbo Integrator (further TI) window I selected ODBC data source type and provide name of the connector (RCAmodelPump) as well as connection credentials. Query window enables to select demanded values. In this case I would like to retrieve all data from table *Products*, therefore I use *select * from products* statement. Result could be seen in the lower part of the window. After defining the source, we can move to the next tab and specify what singular columns will represent. Picture no. 22 shows that for every column (PRODUCT_ID, PRODUCT_NAME, PRODUCT_TYPE) TI created one variable, to which I added one, not stored in DW, called PRODUCTS_ALL. In the 4th column of the variable tab, content of the particular variables have to be determined. As we can see, PRODUCT_ID will be ignored, PRODUCT_NAME column will handle leaf elements of the dimension (therefore selected content - Element), PRODUCT_TYPE will represent consolidated elements (Deposit Account and Credit Account) and added PRODUCT_ALL will include only one, root element of the dimension storing consolidated values for all products. For the better understanding, hierarchy of Product dimension is closely specified in the picture no. 20. Now, definition of the process could be moved into tab *Maps* and sub tab *Dimensions* where we specify name of the dimension (column no.3) and action which will be done after running the process (column no.5) - *recreate* the dimension with actual elements (products). In sub tab *Consolidation*, way how elements are bounded one to another is determined. As we can see in picture no. 24, PRODUCT_TYPE variable will have child elements (column no.3) from PRODUCT_NAME and root element PRODUCT_ALL will be parent for PRODUCT_TYPE. This was the last necessary definition for creating dimension from data source. Now, the process has to be run in order to create dimension.

Column	Dimension Hierarchy				Variable	Content
none	Products				PRODUCT_ALL	Consolidation
PRODUCT_TYPE	Deposit		Credit		PRODUCT_TYPE	Consolidation
PRODUCT_NAME	Personal	Saving	Loan	Morgage	PRODUCT_NAME	Element
PRODUCT_ID	1	2	3	4	PRODUCT_ID	Ignore

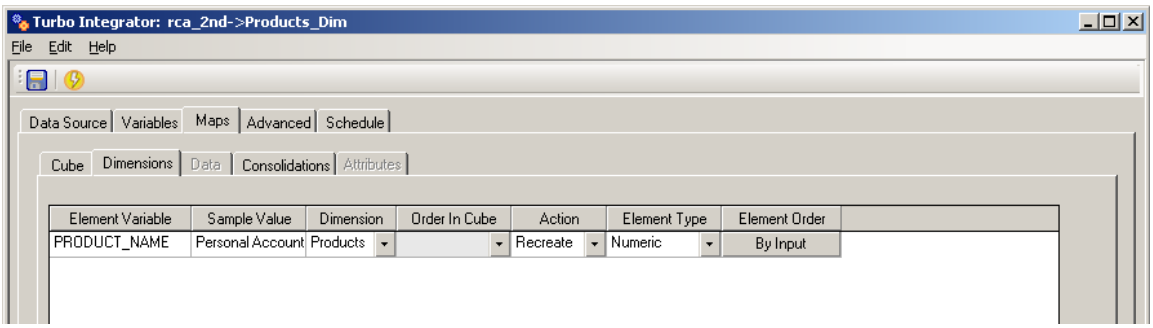
Picture 20: Product dimension detail



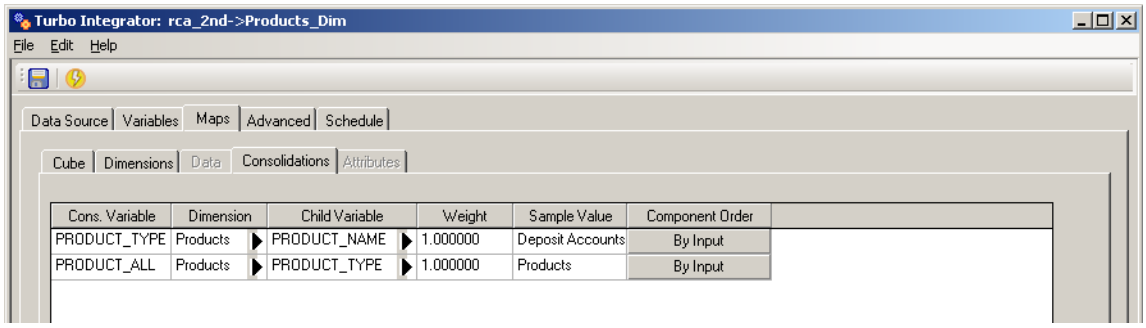
Picture 21: Defining Data Source in Turbo Integrator process



Picture 22: Defining particular columns in Variable tab ob TI process



Picture 23: Definition of dimension name and action



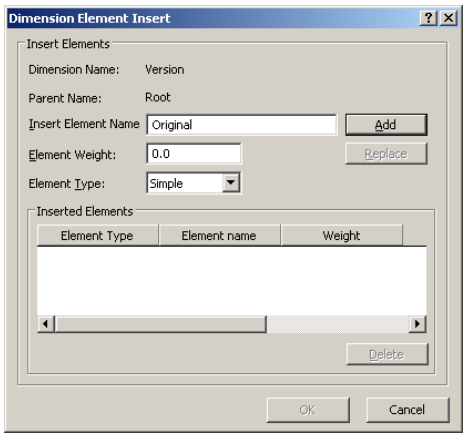
Picture 24: Defining consolidation

8.3.2.2. Time dimension

Time dimension will be created much like Product dimension using TM1 Turbo Integrator process. Therefore we will not describe particular steps again. It includes only for elements (Quartiles), which are sufficient for this thesis, however we predict that client might require bigger detail in the future, like month or days, for which will be process loading more convenient and therefore I selected this approach.

8.3.2.3. Version dimension

Version dimension will initially include only one element (Original), others are expected to be added by user in the future. Therefore I selected creating dimension by hand. Window for adding elements is depicted at picture no. 25, as component of Dimension Editor in which *Original* is being inserted as simple element of *Version* dimension.



Picture 25: Insert new element window in Dimension Editor

8.3.2.4. Sales metrics dimension

As we can observe in the picture no. 14, Sales&Production cube includes one more dimension called Sales metrics. This holds two different values – *count* and *volume*. *Count* element will embrace number of sold products and will be applied to every element of *Products* dimension however *Volume* entity could hold information only about products from

Credit Account product type such as *Loan* or *Mortgage*. This value represents total amount of money lent to debtors in singular quartile through particular credit account product type. Because dimension includes only two elements, it will be as well created by hand.

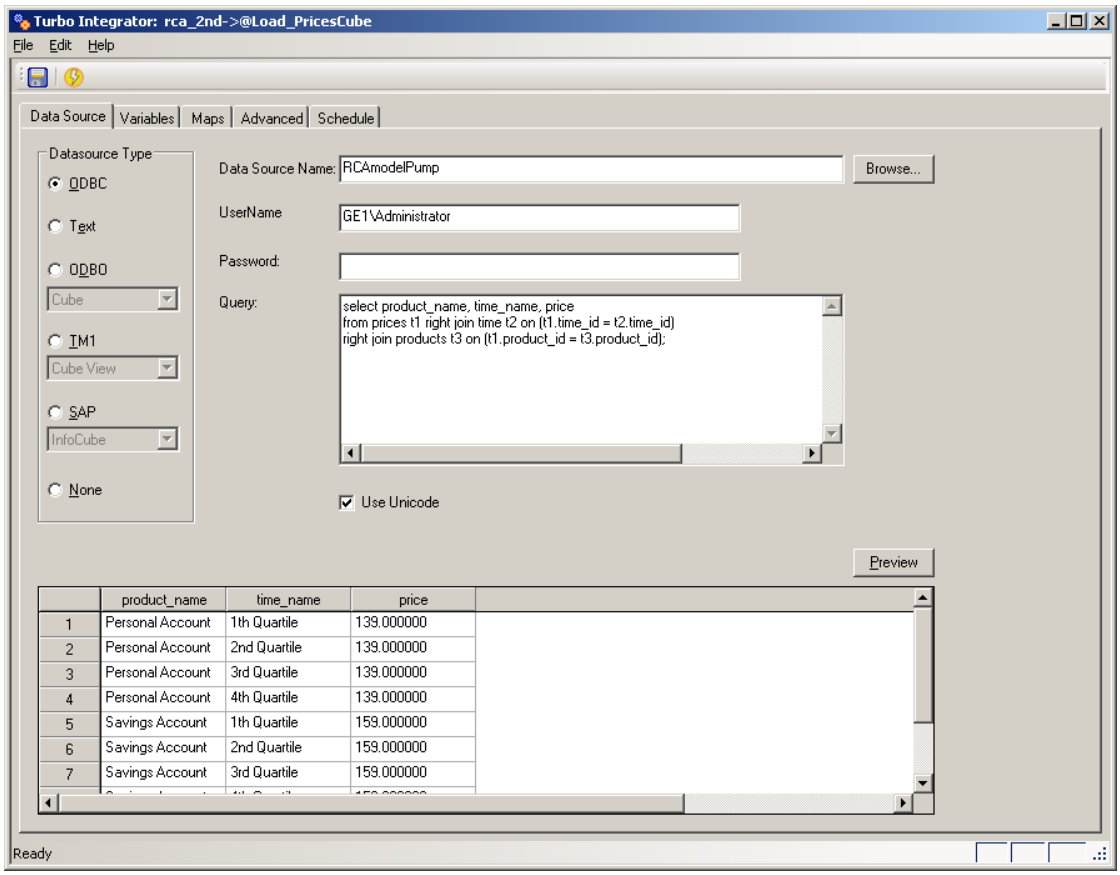
Yet we have created all dimensions required to build up cubes from *Revenue* component and could move to construction of cubes itself.

8.3.2.5. Prices cube

While building *Prices* cube, we are able to define which dimensions will be included and at the same time load values using TM1 process. The process (*Load_PriceCube*) will start with definition of data source holding demanded data. It will be Prices table which consists of three columns – TIME_ID, PRODUCT_ID and PRICE. However we need to map prices to existing elements in *Products* and *Time* dimensions and therefore need to replace ID's in the table with their names. It could be done with following SQL statement:

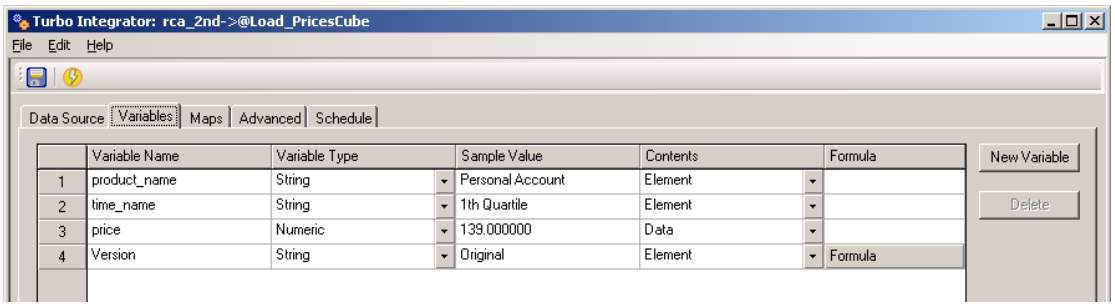
select product_name, time_name, price from prices t1 right join time t2 on (t1.time_id = t2.time_id) right join products t3 on (t1.product_id = t3.product_id);

Output of this query is shown in the lower part of the picture no. 26. This table is sufficient to fill Revenue cube, which enables us to move to the *Variables* tab.



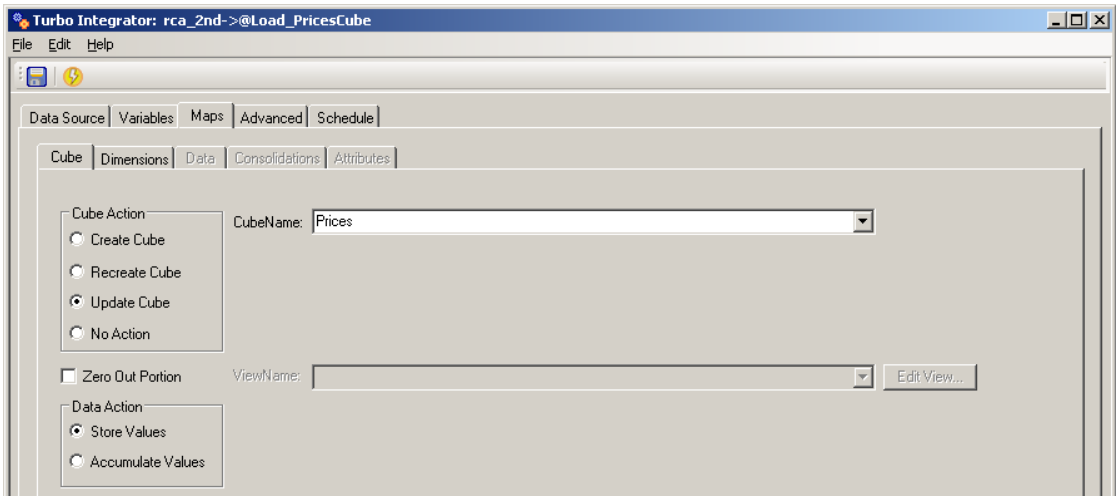
Picture 26: Defining Data Source for filling and creating Prices cube

In *Variables* tab (picture no. 27), content of singular columns in specified as follows: *product_name* column holds elements which will be mapped to Product dimension, therefore content *Element* is selected. Also *time_name* variable will be set to *Element* content and Price column will represents data of the cube, therefore defining *Data* content. Finally I manually inserted variable *Version* which will represent 3rd dimension in the cube and all values will be automatically mapped to its singular element *Original*.



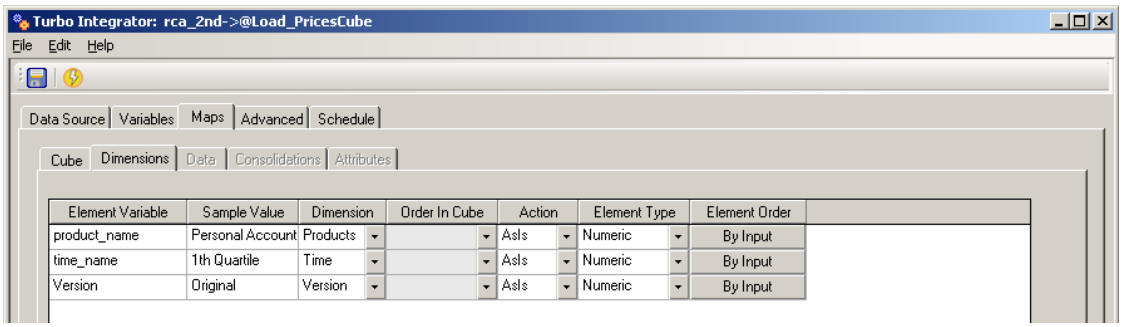
Picture 27: Variables tab of Load_PriceCube process

In the tab *Maps* and sub tab *Cube* (picture no. 28) I selected *update* action (update all values while running the process) and provide name *Prices* for the cube.

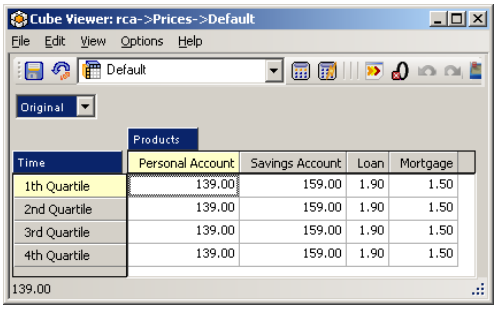


Picture 28: Defining Cube properties

Moving to the sub tab *Dimensions* I selected mapping of the variables to the existing dimensions (column no.3), which means that elements in column PRODUCT_NAME will be mapped to the *Product* dimension, TIME_NAME to the *Time* and all values to the single element of the Version dimension – *Original*. After this, process is ready to be run.



Picture 29: Mapping variables on existing dimensions



Picture 30: Revenue cube

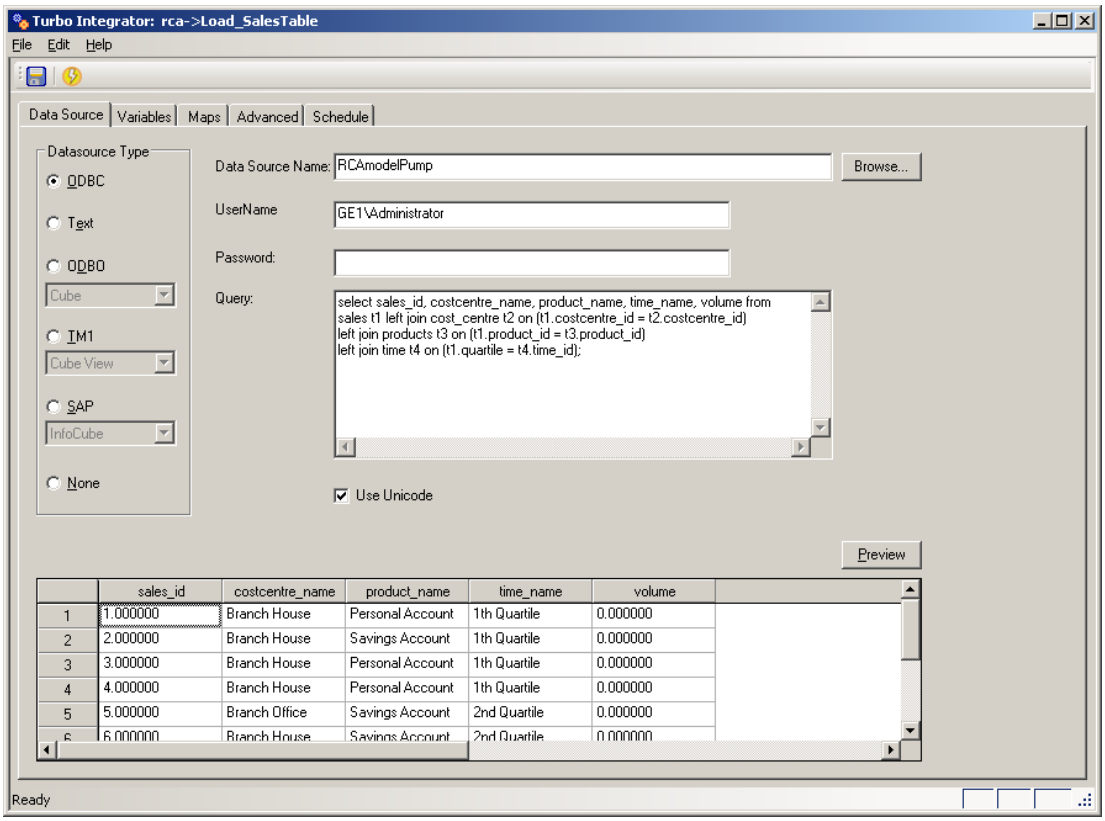
Price Cube with loaded values is depicted at picture no. 30. Both dimensions *Products* and *Time* are filtered to subsets only with elements on lowest level of hierarchy, because consolidated data would not have any predicative value while observing prices.

8.3.2.6. Sales&Production Cube

With *Price* cube developed, we are able to move to *Sales&Production* cube implementation. As was mentioned in architecture and design part of the solution, this cube will hold counts and volumes of sold product types in particular quartiles. Instead of the design, however it was not required, I considered as appropriate to include also *Cost Centers* dimension, which will difference sales to particular branches. Dimension will be created using TM1 processes because of high number of possible elements as well as their probable changes.

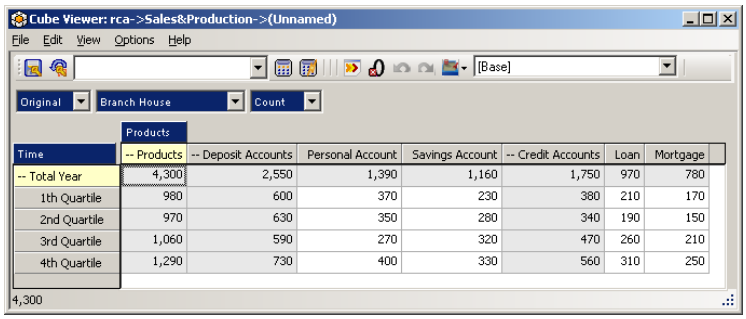
Values will be loaded from the Sales cube stored in the DW. To retrieve required table structure we have to provide following SQL statement:

select sales_id, costcentre_name, product_name, time_name, volume
from sales t1 left join cost_centre t2 on (t1.costcentre_id = t2.costcentre_id) left join products t3 on (t1.product_id = t3.product_id) left join time t4 on (t1.quartile = t4.time_id);

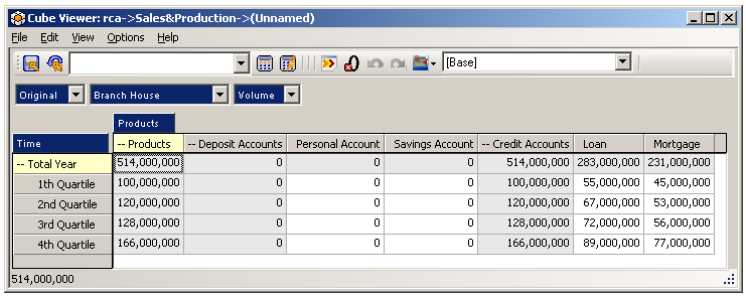


Picture 31: Data source tab for Load SalesTable process

Result table will include all necessary data to build *Sales&Production* Cube. However, instead of the previous cube, it includes two elements in metrics (*sales metrics*) dimension. Therefore we prepared process which will insert for every row +1 to the corresponding cell in *Count* dimension and +volume to the *Volume* dimension. If product is from deposit account and do not have any volume, in *Volume* entity will be accumulate zero, however in *Credit Account* product types will be accumulated total volumes of borrowed money. Result of the accumulation process is shown in pictures no. 32 for *Count* element and in picture no. 33 for *Volume* element.



Picture 32: Slice through Count element in Sales metrics dimension

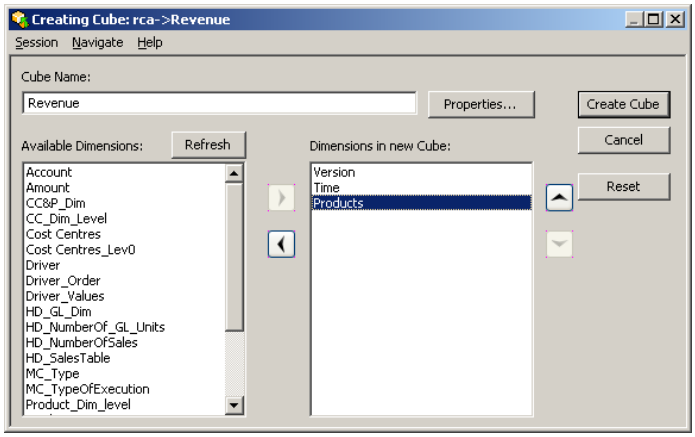


Picture 33: Slice through Volume element in Sales metrics dimension

Having developed both cubes we can create Revenue one and write rule for multiplication of prices and sales to acquire revenue in particular detail.

Revenue cube

Revenue cube will be build up by hand and values will be automatically calculated according to static rule. It will contain three dimensions – *Version, Time and Product.* Window for creating cube is depicted at picture no. 34. I provided name and moved corresponding dimensions from *Available Dimensions* window into *Dimensions in new Cube* column.



Picture 34: Creating Revenue cube

Now calculation rule has to be specified. It will look as follows.

```
[] = N: if (ElIsPar('Products','Deposit Accounts',!Products)=1,
DB('Prices',!Version,!Time,!Products)*DB('Sales&Production',!Version,'Cost Centres','Count',!Time,!Products),
(DB('Prices',!Version,!Time,!Products)*DB('Sales&Production',!Version,'Cost Centres','Volume',!Time,!Products))\100);
```

For better understanding of this advanced rule I will decompose it and will explain singular parts in following syntax boxes.

TM1 rules syntax 1. Structure of the rule and defining calculated cells

Structure of every rule is the same and looks as follows:

What has to be calculated = How it will be calculated

What has to be calculated

User is able to constraint cube to cells for which rule needs to be applied. It could be performed by inserting elements, their values will be calculated into brackets []. As I did not constraint any dimension to some elements, brackets are empty and rule will be applied to every cell in the cube.

N: Statement after equal character

Specifies that rule has to be applied to every cell holding simple (number) elements. If user demand to calculate consolidated ones, he needs to use C: statement or for string cells S.

TM1 rules syntax 2. IF statement

User is able to create conditions by using IF statement. It has following structure.

IF(statement, rule for true result, rule for false result);

! character

If rule is applied to more cells, TM1 engine gradually iterates through all of them. User is able to find out element from specific dimension defining cell which is currently processed by using !dimension_name function. For example, if user determine rule for every simple (N:) cell in Prices cube, in first step !Product returns Personal Account, during second iteration then returns Saving Account etc.

TM1 rules syntax 3. ElIsPar function

Syntax:

ElIsPar(dimension, element 1, element 2)

ELISPAR determines whether element 1 is parent of element 2 in the specified dimension. The function returns TRUE (1) if element 1 is a parent of element 2, otherwise the function returns FALSE (0). [24]

Statement in IF function of Revenue cube rule:

ElIsPar('Products','Deposit Accounts',!Products)=1

evaluates, whether current element of *Products* dimension (*!Products*) is child of 'Deposit Accounts' consolidated elements. If it is true, first part of IF statement is executed, otherwise second one.

[24] TM1 Reference Guide

TM1 rules syntax 4. DB function

If user needs to reach for a value in other cube, he has to use DB statement.

Syntax:

DB(cube name, element of 1st Dimension in the cube, element of 2nd dimension in the cube, ...)

True and false statements of IF condition

If TM1 engine calculates cells defined by Deposit Account product type (ElisPar statement true), it has to calculate revenue as follows:

*DB('Prices',!Version,!Time,!Products)*DB('Sales&Production',!Version,'Cost Centres','Count',!Time,!Products)*

Price of current version, time and product TM1 is calculating multiplied by number of sold products (Count element) in Sales&Production cube for current version, time and product and from all cost centers.

If the engine calculates revenue of Credit Accounts, rule looks as follows:

*(DB('Prices',!Version,!Time,!Products)*DB('Sales&Production',!Version,'Cost Centres','Volume',!Time,!Products))\100);*

Revenue is calculated as multiplication of interest rate stored in *Price* cube and volume of borrowed money from *Sales&Production* cube in *Volume* element. Interest rate is stored in percentage, therefore we need to divide result by 100 in order to retrieve actual amount.

8.3.3. Variable costs component

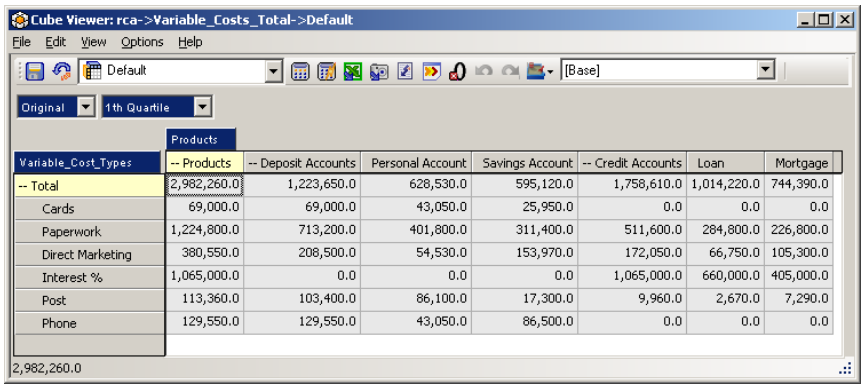
After explaining different procedures and functions in previous paragraph, variable costs component will be very simple to build. Therefore, I will dedicate to this component just a little discussion and move forward to much more difficult and interesting *Fixed costs allocation* component. User firstly needs to construct *variable costs metrics* dimension, by inserting particular elements by hand. Then he is able to build up *Variable cost per unit* cube by defining its name and included dimension. According to architecture and design part of the solution, they will be *Version, Time, Products* and *Variable costs metrics*. Total values will be however calculated in *Variable costs Total* and will include same dimensions. As was mentioned in theoretical part of the thesis, variable costs originate with every produced output and therefore *Variable costs Total* cube will be calculated by multiplying Variable *costs per Unit* and *Sales&Production* cube, where count of sold products is hold.

Variable costs Total rules

['Interest %']=N:DB('Variable Costs per Unit',!Version,'Interest %',!Time,!Products) DB('Sales&Production',!Version,'Cost Centres','Volume',!Time,!Products);*

[]=N:DB('Variable Costs per Unit',!Version,!Variable_Cost_Types,!Time,!Products) DB('Sales&Production',!Version,'Cost Centres','Count',!Time,!Products);*

First rule calculates variable cost *Interest*, which depends on volume of required money to be lent. Any other metric is then calculated by multiplying its unit cost and count.



Variable_Cost_Types	-- Products	-- Deposit Accounts	Personal Account	Savings Account	-- Credit Accounts	Loan	Mortgage
-- Total	2,982,260.0	1,223,650.0	628,530.0	595,120.0	1,758,610.0	1,014,220.0	744,390.0
Cards	69,000.0	69,000.0	43,050.0	25,950.0	0.0	0.0	0.0
Paperwork	1,224,800.0	713,200.0	401,800.0	311,400.0	511,600.0	284,800.0	226,800.0
Direct Marketing	380,550.0	208,500.0	54,530.0	153,970.0	172,050.0	66,750.0	105,300.0
Interest %	1,065,000.0	0.0	0.0	0.0	1,065,000.0	660,000.0	405,000.0
Post	113,360.0	103,400.0	86,100.0	17,300.0	9,960.0	2,670.0	7,290.0
Phone	129,550.0	129,550.0	43,050.0	86,500.0	0.0	0.0	0.0

2,982,260.0

Picture 35: Variable costs Total cube

8.3.4. Fixed costs allocation component

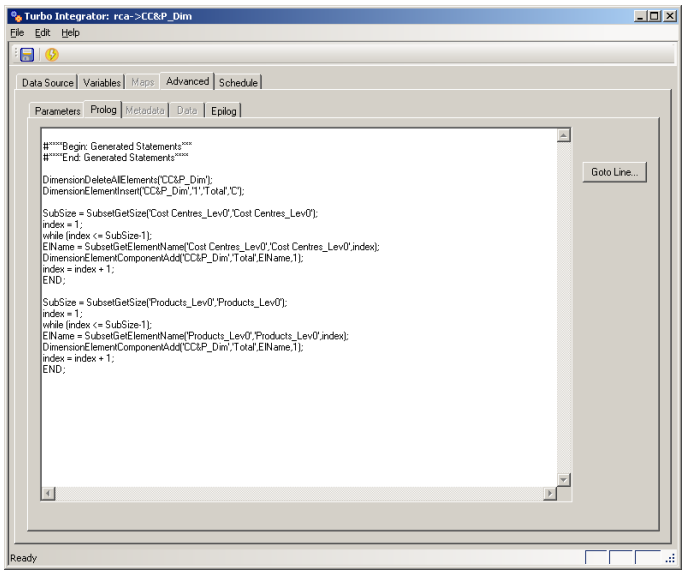
Fixed costs allocation component is undoubtedly the most difficult one, therefore I selected approach in which flexible cost driver's management and assignment will be solved firstly, followed by the allocation engine itself. *Drivers Package* cube seems to be the good starting point for the development. As designed in architecture part of the solution, it contains *Drivers* and *CC&P* dimensions.

8.3.4.1. CC&P Dimension

CC&P dimension originates as combination of *Cost Centers* and *Product* dimension which I have already created. Because I want to keep this dimension updated with every change in elements of *Cost Centers* or *Product*s dimension, it is convenient to create it with a process. However this process will contain advanced functions which are not included in tabs we used so far, therefore I have to write them by hand in *Advanced* tab of Turbo Integrator window. In the picture no. 36 is depicted whole process script. Because I am not working with any data source, sub tabs *Data* and *Metadata* are disabled. Considering the fact, that process is very simple, I wrote it for better comprehension and orientation completely to sub tab *Prolog*. Now, let's examine particular lines or procedure.

Process starts at line 5, which ensures that all elements will be deleted before recreating dimension.

```
DimensionDeleteAllElements('CC&P_Dim');
```



Picture 36: Advanced tab of CC&P_dim process

It is followed by inserting first element called *Total* at first place of CC&P dimension. This will be consolidated element (C) and will contain summed value of all children elements which will be inserted further.

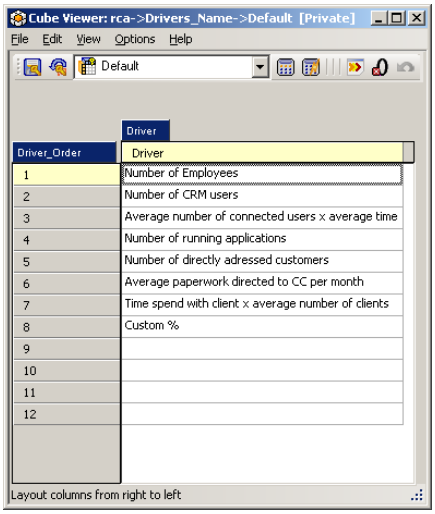
```
DimensionElementInsert('CC&P_Dim','1','Total','C');
```

Next step will be adding leaf elements from Cost Centers (Cost Centers_Level0) and Products (Products_Level0) dimensions[25]. For both dimensions I created simple procedures which find out number of entities in dimension and for each (except last – consolidated Total) will retrieve its name and add it as child (component of Total element) into CC&P dimension. Syntax is very comprehensive and do not need to be closely explained.

```
SubSize = SubsetGetSize('Cost Centres_Lev0','Cost Centres_Lev0');
index = 1;
while (index <= SubSize-1);
ElName = SubsetGetElementName('Cost Centres_Lev0','Cost Centres_Lev0',index);
DimensionElementComponentAdd('CC&P_Dim','Total',ElName,1);
index = index + 1;
END;
```

8.3.4.2. Driver's Name cube

With *CC&P* dimension created we are able to move to the second one of the Drivers Package cube called Drivers. This dimension will hold current offering of available cost drivers. However it is demanded to be managed (add or remove drivers) by end user, therefore we need to prepare an easy interface for their administration, in order to avoid necessity of accessing TM1 Server Explorer when user require to adapt them. This could be solved by creating simple cube with some cells in which is user able to manage driver's names. Cube will be called *Drivers_Name* and for this implementation will include 12 cells as depicted in picture no. 37.



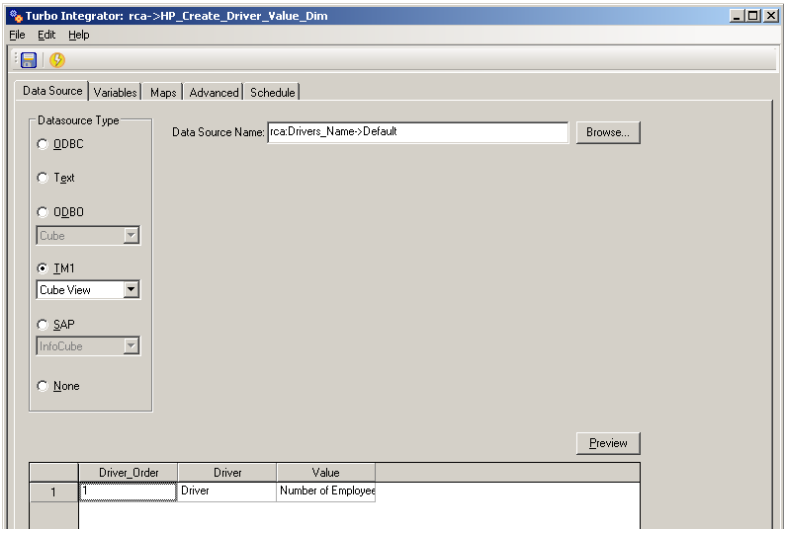
Picture 37: Drivers Name cube

8.3.4.3. Driver's Value dimension

We also need to create process which will handle values in cube and generate from them Drivers_Value dimension. As data source I now selected TM1 Cube Drivers_Name with default view. In the lower part

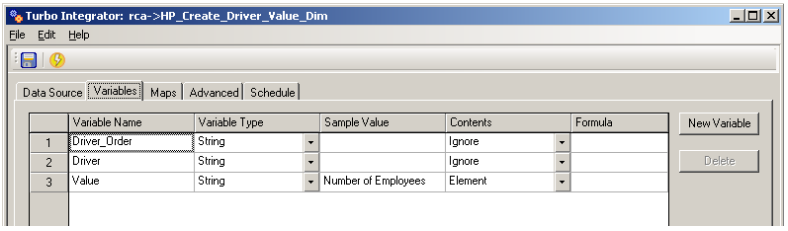
[25] Same way I am creating CC&P dimension I constructed before _Level0 dimensions and subsets from both Cost Centers and Products dimensions. These include only leaf elements of their structure and one consolidated called Total. This is appropriate when working with indexed and when user wants to avoid obstacles with consolidated elements. When loading dimensions we do not know, on which positions will be consolidated elements such as Deposit Accounts or Production Centers. However when we always update _Level0 dimension immediately after recreating primal one, we can avoid possible complications while working with baffling dimensions.

of the picture no. 38 is previewed how cube was pulled apart into table with three columns (2 dimensions and values).



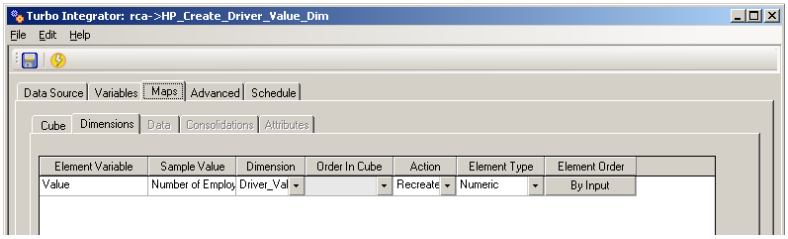
Picture 38: Defining data source for Drivers_Value dimension

For every column TI generated one variable. Although I am interested only in values of the cube, I select *ignore* for both variables representing dimensions of the cube and for *Values* variable select *Element* content (Picture no.39). Further I have to determine *Mapping* preferences.



Picture 39: Variables tab of Create Drivers Value dimension process

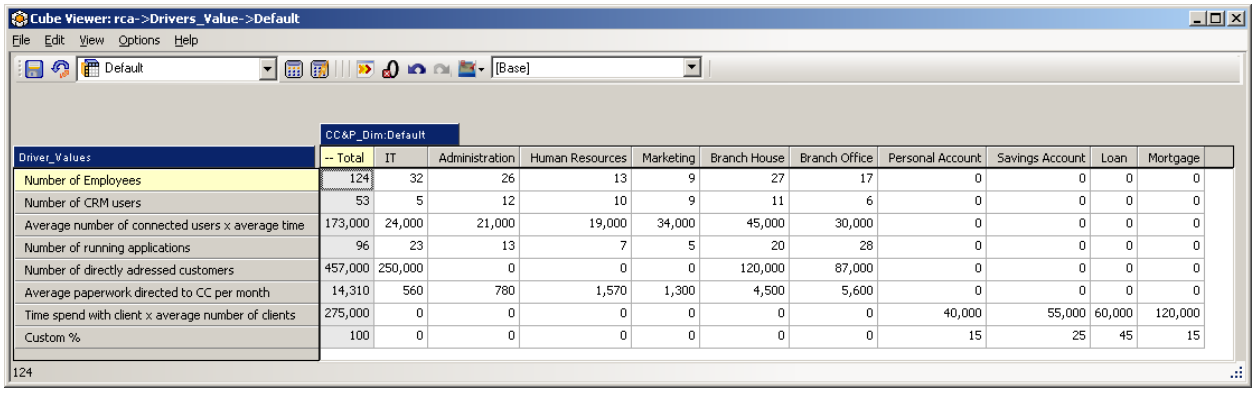
In sub tab *Dimensions* of *Maps* tab I consequently inserted name of the new dimension I require to build. This process needs to be run after every change to *Drivers_Name* cube user requires to be applied. In order to support appropriate interface, process will be bounded to button *Execute changes* next to the inserting cube while constructing Excel based user interface sheets.



Picture 40: Defining name of Drivers Value dimension

8.3.4.4. Drivers Package

With both dimensions created as well as processes which ensure their update, we can create so called *Drivers Package* handling values for every driver. These could be filled from different sources as we discussed before, however for this implementation we consider their filling by hand. Slice of the cube, which I called *Drivers_Value* and which could be easily managed is depicted at picture no. 41.



Cube Viewer: rca->Drivers_Value->Default

File Edit View Options Help

Default [Base]

Driver_Values	CC&P_Dim:Default										
	-- Total	IT	Administration	Human Resources	Marketing	Branch House	Branch Office	Personal Account	Savings Account	Loan	Mortgage
Number of Employees	124	32	26	13	9	27	17	0	0	0	0
Number of CRM users	53	5	12	10	9	11	6	0	0	0	0
Average number of connected users x average time	173,000	24,000	21,000	19,000	34,000	45,000	30,000	0	0	0	0
Number of running applications	96	23	13	7	5	20	28	0	0	0	0
Number of directly adressed customers	457,000	250,000	0	0	0	120,000	87,000	0	0	0	0
Average paperwork directed to CC per month	14,310	560	780	1,570	1,300	4,500	5,600	0	0	0	0
Time spend with client x average number of clients	275,000	0	0	0	0	0	0	40,000	55,000	60,000	120,000
Custom %	100	0	0	0	0	0	0	15	25	45	15

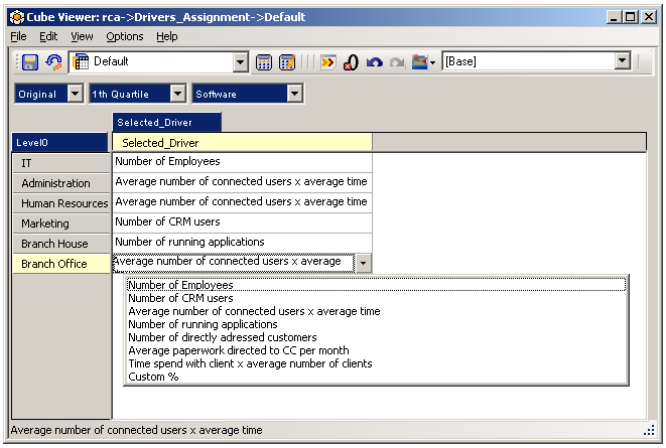
124

Picture 41: Drivers Value cube

8.3.4.5. Drivers Assignment

Next step in *Fixed costs allocation component* will be after defining drivers their assignment to particular amounts (Fixed costs) we would like to allocate. Architecture and design of the cubes reveals that client would like to have possibility to allocate every expenditure in different time, on different account and originated in different cost centers according to various drivers. Considering this fact, *Drivers_Assignment* cube will include four main dimensions: *Version*, *Time*, *Account* and *Cost Centers*. We have not yet constructed Account dimension, however it will be loaded from GL table much like other dimensions by using processes.

Fifth dimension of the cube will be metrics dimension, which will include only one – *Selected_Driver* element. Because we would like to enable end user comfortably select driver for every type of expenditure through pick list of available options, *Selected_Driver* element has to be string type. Restriction will be set by using attribute called *picklist* and offered values will be defined by dimension which elements should appear in the list.



Cube Viewer: rca->Drivers_Assignment->Default

File Edit View Options Help

Default [Base]

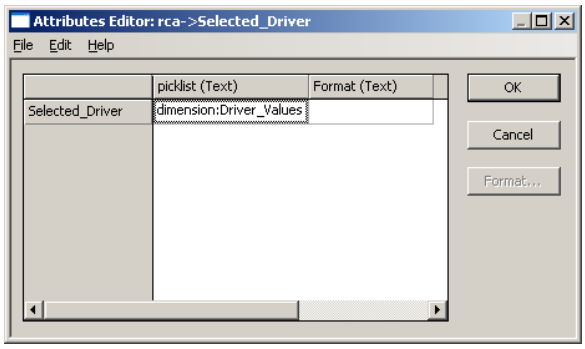
Original 1th Quartile Software

Level0	Selected_Driver
IT	Number of Employees
Administration	Average number of connected users x average time
Human Resources	Average number of connected users x average time
Marketing	Number of CRM users
Branch House	Number of running applications
Branch Office	Average number of connected users x average

Number of Employees
Number of CRM users
Average number of connected users x average time
Number of running applications
Number of directly adressed customers
Average paperwork directed to CC per month
Time spend with client x average number of clients
Custom %

Average number of connected users x average time

Picture 42: Drivers Assignment cube with picklist on every cell



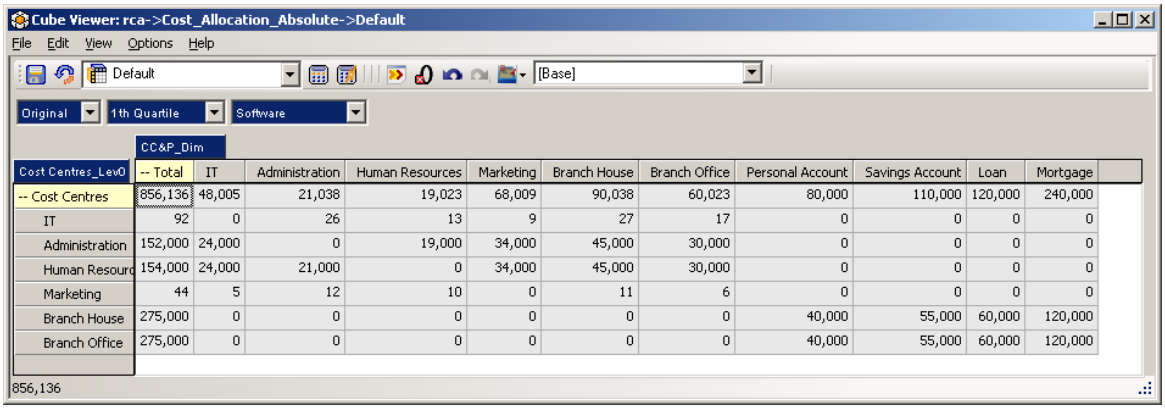
Picture 43: Deffining picklist values

8.3.4.6. Cost Allocation_Absolute

According to the selected drivers, values have to he transferred into Cost Aloocation_Absolute cube, where will be further processes before particular allocation. Architecture of the application determines five dimensions in this cube: *Cost Centers* (allocation will run only among elements on lowest level of hierarchy, therefore more convenient to use *Cost Centers_Level0* dimension to avoid obstacles while working with elements indexes), *CC&P*, *Version*, *Time* and *Account*. They will be inserted and cube will be created by hand, as well as will be written static rule for data transfer from Drivers_Values cube according to assignment, which will looks as follows:

```
[]=N:DB('Drivers_Value',DB('Drivers_Assignment',!Version,!Time,!Account,!Cost Centres_Lev0,'Selected_Driver'),!CC&P_Dim);
```

Rule's syntax is simple however reader should notice how was instead of element name nested reference to external cube. This enables values loading according to selected driver in *Drivers Assignment* cube.



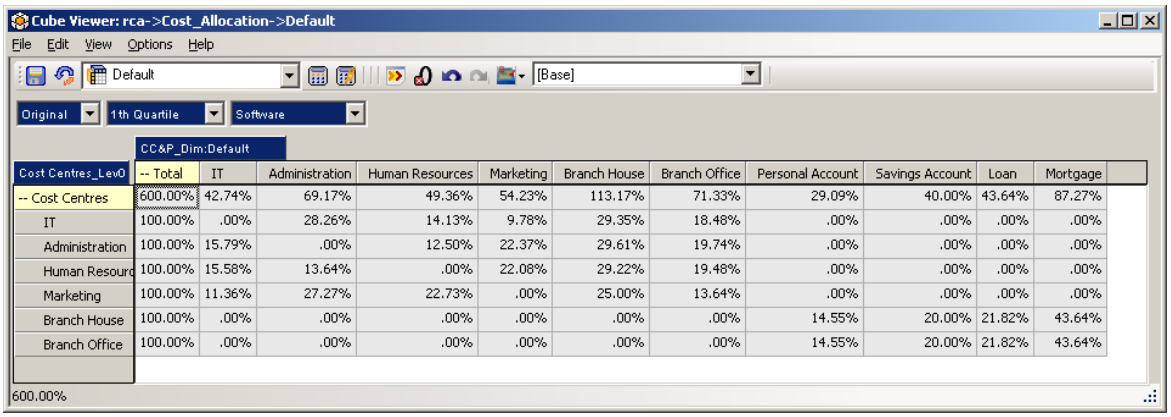
Cost Centres_Lev0	-- Total	IT	Administration	Human Resources	Marketing	Branch House	Branch Office	Personal Account	Savings Account	Loan	Mortgage
-- Cost Centres	856,136	48,005	21,038	19,023	68,009	90,038	60,023	80,000	110,000	120,000	240,000
IT	92	0	26	13	9	27	17	0	0	0	0
Administration	152,000	24,000	0	19,000	34,000	45,000	30,000	0	0	0	0
Human Resourc	154,000	24,000	21,000	0	34,000	45,000	30,000	0	0	0	0
Marketing	44	5	12	10	0	11	6	0	0	0	0
Branch House	275,000	0	0	0	0	0	0	40,000	55,000	60,000	120,000
Branch Office	275,000	0	0	0	0	0	0	40,000	55,000	60,000	120,000

Picture 44: View on Cost Allocation Absolute Cube

8.3.4.7. Cost Allocation

Allocation process however requires working with relative values (drivers), therefore we need to recalculate *Cost Allocation_Absolute* cube to *Cost Allocation* which operates with percentages. This will be as well ensured by following rule.

```
[]=N:DB('Cost_Allocation_Absolute',!Version,!Time,!Account,!Cost Centres_Lev0,!CC&P_Dim)\
DB('Cost_Allocation_Absolute',!Version,!Time,!Account,!Cost Centres_Lev0,'Total');
```



Cost Centres_Lev0	-- Total	IT	Administration	Human Resources	Marketing	Branch House	Branch Office	Personal Account	Savings Account	Loan	Mortgage
-- Cost Centres	600.00%	42.74%	69.17%	49.36%	54.23%	113.17%	71.33%	29.09%	40.00%	43.64%	87.27%
IT	100.00%	.00%	28.26%	14.13%	9.78%	29.35%	18.48%	.00%	.00%	.00%	.00%
Administration	100.00%	15.79%	.00%	12.50%	22.37%	29.61%	19.74%	.00%	.00%	.00%	.00%
Human Resourc	100.00%	15.58%	13.64%	.00%	22.08%	29.22%	19.48%	.00%	.00%	.00%	.00%
Marketing	100.00%	11.36%	27.27%	22.73%	.00%	25.00%	13.64%	.00%	.00%	.00%	.00%
Branch House	100.00%	.00%	.00%	.00%	.00%	.00%	.00%	14.55%	20.00%	21.82%	43.64%
Branch Office	100.00%	.00%	.00%	.00%	.00%	.00%	.00%	14.55%	20.00%	21.82%	43.64%

Picture 45: Cost Allocation cube with percentage values

8.3.4.8. Amounts for Allocation

Further cube I will construct before implementing allocation process itself will be cube holding values which have to be allocated. According to application architecture it will include four main dimensions plus dimension holding remaining values in every step of process. Steps dimensions is possible to incrementally enlarge every loop, however this requires more advanced implementation, and thesis restricts its detail description. Therefore I choose solution, where elements in *Steps* dimension are prepared beforehand. For demonstration of this implementation will be 20 elements sufficient. Sixth dimension I decided to include is also metrics dimension called *Amount* which holds in element *Amounts* remaining costs for allocation and in element *Difference*, values which were reallocated to products in particular step. This is implemented as subtraction of amount in current step and amount from previous one.

```
['1','Difference']=0;
['Difference']=['Amount']-DB('AmountsForAllocation',!Version,!Time,!Account,DIMNM('Steps',(DIMIX('Steps',!Steps)-1)),!Cost Centres,'Amount');
```

Cube Viewer: rca->AmountsForAllocation->Default

File Edit View Options Help

Default [Base]

Original 1th Quartile Software

Amount	Cost Centres	Steps 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Amount	Branch Office	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Branch House	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-- Production Centre	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Marketing	1,000,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Human Resources	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Administration	300,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	IT	600,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-- Support Centre	1,900,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-- Cost Centres	1,900,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Difference	Branch Office	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Branch House	0	0	0	0	0	0	0	0	0	0	-0	0	0	0	0	0	0	0	0	0
	-- Production Centre	0	0	0	0	0	0	0	0	0	0	-0	0	0	0	0	0	0	0	0	0
	Marketing	0	-1,000,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Human Resources	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Administration	0	-300,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	IT	0	-600,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-- Support Centre	0	-1,900,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-- Cost Centres	0	-1,900,000	0	0	0	0	0	0	0	0	-0	0	0	0	0	0	0	0	0	0

0

Picture 46: Amounts for Allocation after loading fixed costs and before allocation

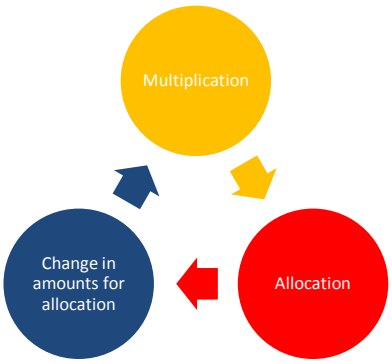
Initial values from DW for allocation are loaded with process and are inserted into element 1 in Steps dimension, element Amounts in Amount dimension and corresponding elements of another ones. Result of the load do not markedly differ from another processes descried before in the implementation and could be seen in picture no 46.

8.3.4.9. Multiplied

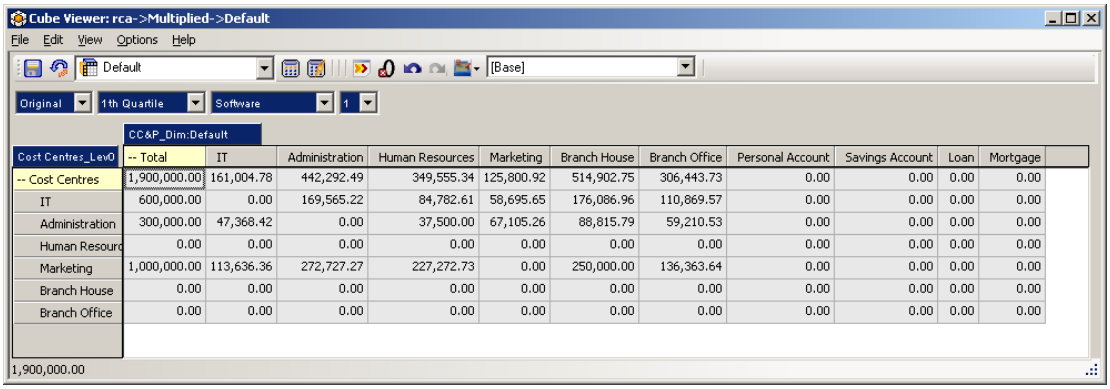
Multiplied is key cube of whole process. Not only it handles values with which operates allocation process, furthermore user is able to find here many interesting data such as who allocated to who (or what) and how much in every step. Values originate from multiplication of *amounts for allocation* with corresponding *cost allocation* percentage. Rule which calculates these values is shown in box further.

```
[]=N:DB('Cost_Allocation',!Version,!Time,!Account,!Cost Centres_Lev0,!CC&P_Dim)*
DB('AmountsForAllocation',!Version,!Time,!Account,!Steps,!Cost Centres_Lev0,'Amount');
```

After the multiplication process looks to the relevant cells and displace their values to appropriate place (modify amounts for allocation – subtract amount which was allocated to other entities and add value which was allocated back) basis for rule will be changed, what causes that rule for multiplication will calculate new values to be processed and write it to next step element. This runs until selected condition is fulfilled. Cycle could be visualized as follows.



Picture 47: Allocation process visualization



Cube Viewer: rca->Multiplied->Default

File Edit View Options Help

Default [Base]

Original 1th Quartile Software 1

CC&P_Dim:Default

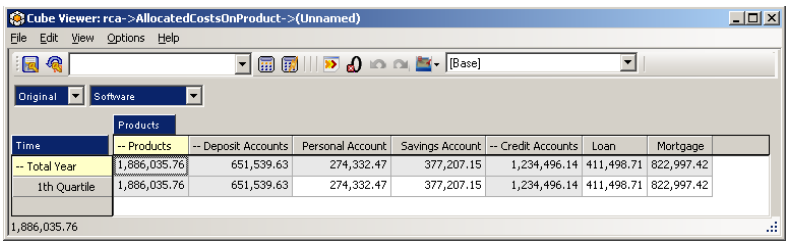
Cost Centres_Lev0	-- Total	IT	Administration	Human Resources	Marketing	Branch House	Branch Office	Personal Account	Savings Account	Loan	Mortgage
-- Cost Centres	1,900,000.00	161,004.78	442,292.49	349,555.34	125,800.92	514,902.75	306,443.73	0.00	0.00	0.00	0.00
IT	600,000.00	0.00	169,565.22	84,782.61	58,695.65	176,086.96	110,869.57	0.00	0.00	0.00	0.00
Administration	300,000.00	47,368.42	0.00	37,500.00	67,105.26	88,815.79	59,210.53	0.00	0.00	0.00	0.00
Human Resourc	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Marketing	1,000,000.00	113,636.36	272,727.27	227,272.73	0.00	250,000.00	136,363.64	0.00	0.00	0.00	0.00
Branch House	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Branch Office	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

1,900,000.00

Picture 48: Multiplied cube with calculated values

8.3.4.10. Allocated Costs on Product

Besides relocating values among Amount for allocation cube cells, allocation process has to ensure storing values which were calculated to products and will not be further processed. This leads to constructing additional cube called *Amounts for allocation* which will store accumulated amounts assigned to particular products. Sliced view through Software element and subset for 1th Quartile, values we were observing in screen shots of cubes so far is depicted at picture no. 49.



Cube Viewer: rca->AllocatedCostsOnProduct->(Unnamed)

File Edit View Options Help

[Base]

Original Software

Products

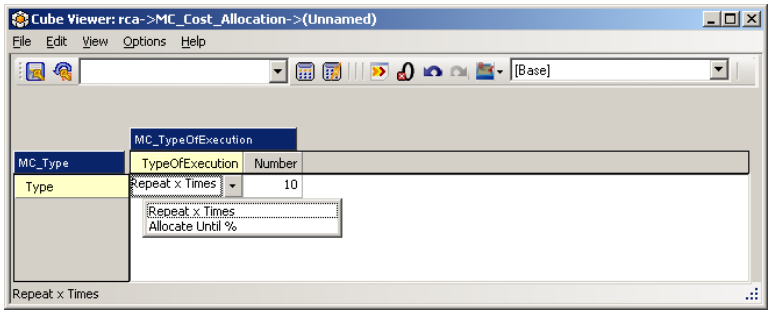
Time	-- Products	-- Deposit Accounts	Personal Account	Savings Account	-- Credit Accounts	Loan	Mortgage
-- Total Year	1,886,035.76	651,539.63	274,332.47	377,207.15	1,234,496.14	411,498.71	822,997.42
1th Quartile	1,886,035.76	651,539.63	274,332.47	377,207.15	1,234,496.14	411,498.71	822,997.42

1,886,035.76

Picture 49: Allocated Costs on Product cube

8.3.4.11. Manager Console for Cost Allocation

Last point of the architecture and design of the solution was requirement to implement different conditions for allocation process such as number of iterations or percentage of allocated costs. End user satisfaction and easefulness could be accomplished with simple cube holding two values – selected condition and appropriate number. Cube view as well as implemented pick list for *Type of Execution* element is depicted at picture no. 50.

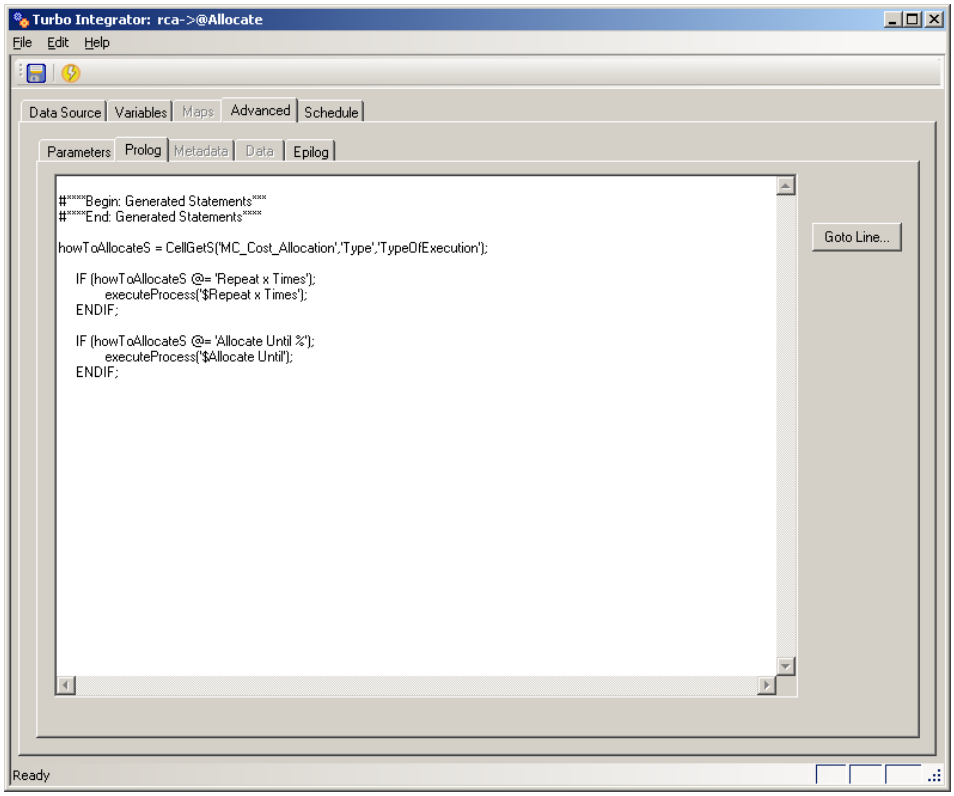


Picture 50: Manages Console

8.3.4.12. Allocate process

Having all required cubes implemented, we are able to move to defining allocation process. Firstly I prepare script (@Allocate process)[26] which will decide what method (process) will be executed. As we can see in picture no.51, I loaded selected value from pick list from *MC_Cost_Allocation* (Manager Console for Cost Allocation) into *howToAllocateS* variable. Following IF statement then decides which process ought to be executed. Processes look the same, they differ only in condition which stops the calculation, and therefore I describe only *Repeat X Times* process in detail. *Allocate Until %* process's body will look the same.

[26] My convention is to name processes which execute another processes with @ as starting character



Picture 51: Advanced tab of @Allocate process

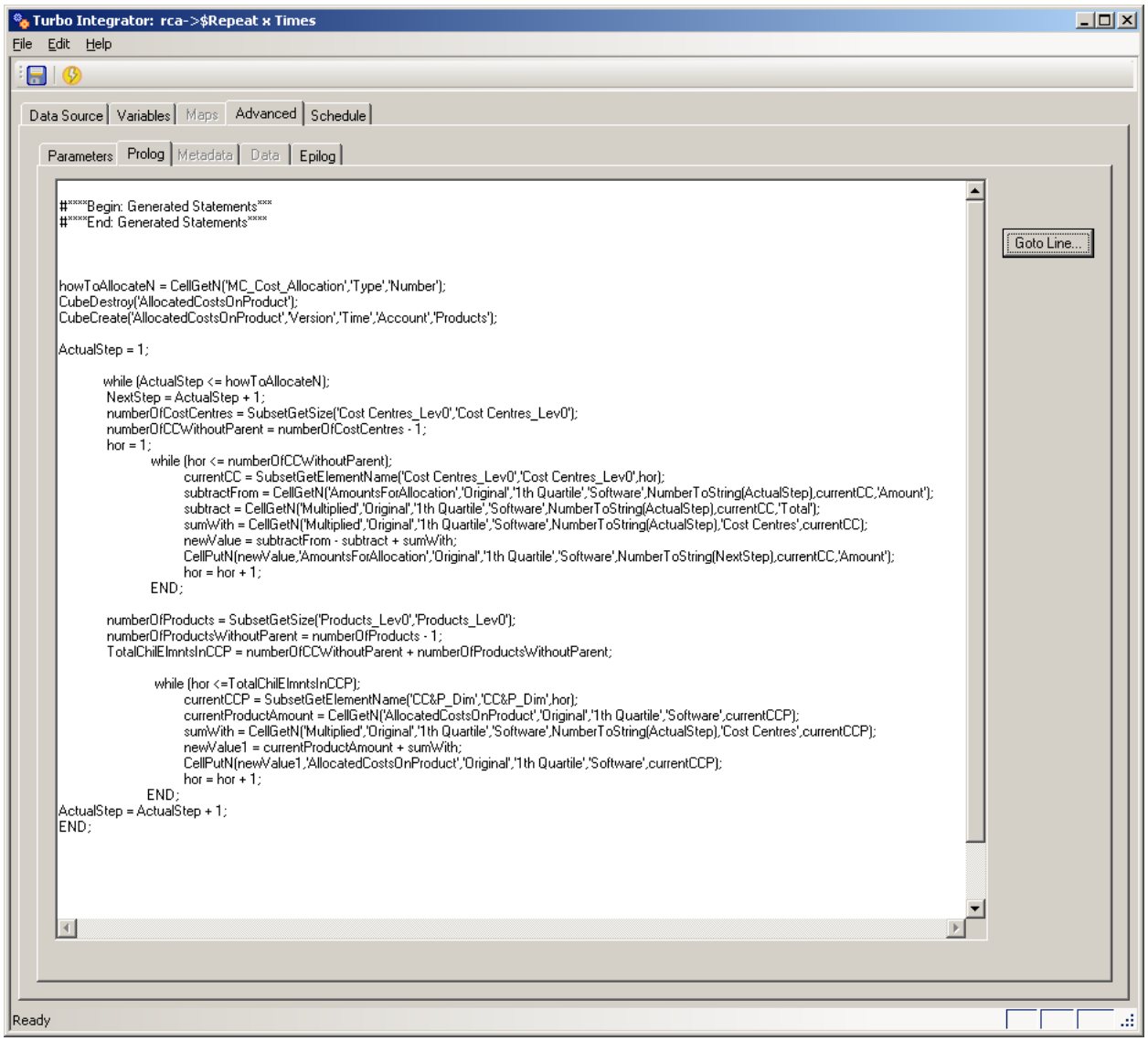
8.3.4.13. Repeat X Times process

In first three lines (picture no. 52) I fill variable howToAllocateN with number from Managers Console, which determines how many loops process has to execute and further deletes and then de novo recreates *AllocatedCostsOnProduct* cube which set to zero all cells.

```
howToAllocateN = CellGetN('MC_Cost_Allocation','Type','Number');
CubeDestroy('AllocatedCostsOnProduct');
CubeCreate('AllocatedCostsOnProduct','Version','Time','Account','Products');
```

Actual step variable will represent current loop of allocation and will be set to 1 at the beginning. It will also correspond with *Steps* dimension elements we will need to use further. Because user selected fixed number of allocation, we will start implementation of *while cycle* with condition allowing running until current loop is lower or equal than defined total steps in *howToAllocateN* variable. Before ending cycle at the last line of the process, we increment Actual step variable to avoid circular reference. *Next step* variable in sixth line of process will hold number pointing to current +1 element in Steps dimension where we would like to write allocated values. Before explaining following lines, let's revive table from architecture and design part of the solution. Red cells in picture no.53 hold values which were allocated out of current Cost Center and we need to subtract them from fixed costs in Amounts for Allocation cube (picture no. 54, element Step 1). This operation will require sub process which will treat every cell

distinctive. To determine number of iterations in this process, we need to find out current number of cost centers and subtract one element *Total*.



Picture 52: Advanced tab of Repeat X times allocation process

Script:

```
numberOfCostCentres = SubsetGetSize('Cost Centres_Lev0','Cost Centres_Lev0');
numberOfCCWithoutParent = numberOfCostCentres - 1;
```

While Cost Centers is horizontal dimension, I called indicator which will point to singular elements of dimension during the sub process as *hor*. We will set it to 1 and define condition for repeating process until reach last leaf element. To the *CurrentCC* variable will be loaded name of current element. It is important to remark, that for simplicity I do not proceed through *Version*, *Time* and *Account* dimension, however define them as static elements: *Original*, *1th Quartile* and *Software*. This markedly simplifies

whole process. In following lines I load into *substractFrom* fixed costs from *Amounts for Allocation* which were in particular Cost Center before allocation (Actual step). Into *subtract* variable is handled value which will be reallocated out of the current Center. Much like some money is allocated out during the process, some are calculated back from other Centers. These are represented by green cells in picture no. 53 and will be stored in *sumWith* variable. *NewValue* variable will handle amount of fixed costs which will remain in Cost Centers after allocation and will be placed back into Amount for allocation cube, however into following step represented by Next Step variable. Whole script then looks as follows:

```
while (hor <= numberOfCCWithoutParent);
currentCC = SubsetGetElementName('Cost Centres_Lev0','Cost Centres_Lev0',hor);
subtractFrom =CellGetN('AmountsForAllocation','Original','1th
Quartile','Software',NumberToString(ActualStep),currentCC,'Amount');
subtract = CellGetN('Multiplied','Original','1th Quartile','Software',NumberToString(ActualStep),currentCC,'Total');
sumWith = CellGetN('Multiplied','Original','1th Quartile','Software',NumberToString(ActualStep),'Cost Centres',currentCC);
newValue = subtractFrom - subtract + sumWith;
CellPutN(newValue,'AmountsForAllocation','Original','1th Quartile','Software',NumberToString(NextStep),currentCC,'Amount');
hor = hor + 1;
END;
```

The same procedure will be executed for the yellows cells in the picture no. 53 which represents allocated values finally on products. Firstly we will find out number of products without consolidated element in *Products* dimension. Sum it with number of cost centers and determine conditions for second sub process. It will continue with *hor* variable, however now will move on vertical dimension. For every product I will find out amount which is currently allocated on particular product in allocated Costs on Produsts as well as value which has to be added. Then sum it in *newValue1* variable and replace current amount with summed one. The whole script looks as follows:

```
while (hor <= numberOfCCWithoutParent);
currentCC = SubsetGetElementName('Cost Centres_Lev0','Cost Centres_Lev0',hor);
subtractFrom = CellGetN('AmountsForAllocation','Original','1th
Quartile','Software',NumberToString(ActualStep),currentCC,'Amount');
subtract = CellGetN('Multiplied','Original','1th Quartile','Software',NumberToString(ActualStep),currentCC,'Total');
sumWith = CellGetN('Multiplied','Original','1th Quartile','Software',NumberToString(ActualStep),'Cost Centres',currentCC);
newValue = subtractFrom - subtract + sumWith;
CellPutN(newValue,'AmountsForAllocation','Original','1th Quartile','Software',NumberToString(NextStep),currentCC,'Amount');
hor = hor + 1;
END;
```

	Cost centers				Products				Total
Cost centers		125 000	25 000	100 000					250 000
	50 000		50 000						100 000
	30 000				90 000	90 000	90 000		300 000
							300 00	300 000	600 000
Total	80 000	125 000	75 000	100 000	90 000	90 000	390 000	300 000	1 250 000

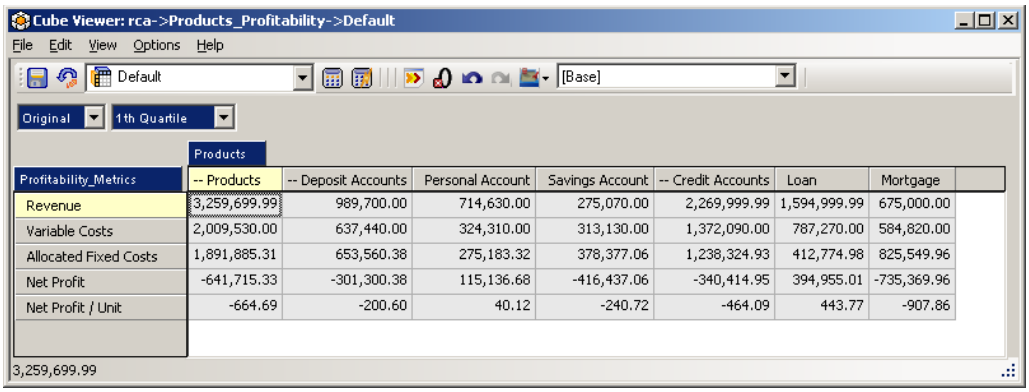
Picture 53: Representation of allocation process

	Amounts	
	Step 1	Step 2
Cost centers	250 000	80 000
	100 000	125 000
	300 000	75 000
	600 000	100 000

Picture 54: Amounts for Allocation cube representation

8.3.4.14. Products Profitability

Result of the application will be available through *Products Profitability* cube, which will aggregate all calculated metrics and provide information about profitability of products as well as profitability per unit including allocated fixed costs. Cube will contain main dimensions we defined as key at the beginning of the architecture and design part of the solution supplemented with metrics dimension including amount from all components such as revenue and expenditures represented by variable and fixed costs. Subtraction will further provide total profitability of products and divided by quantity detail values per unit. Result could be observed in following picture no. 55.



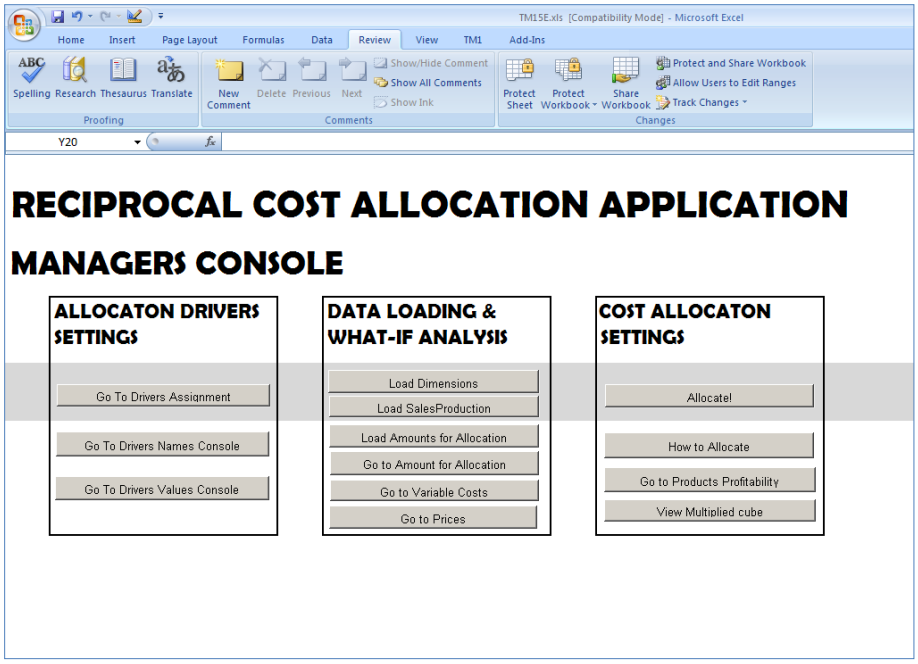
Profitability_Metrics	-- Products	-- Deposit Accounts	Personal Account	Savings Account	-- Credit Accounts	Loan	Mortgage
Revenue	3,259,699.99	989,700.00	714,630.00	275,070.00	2,269,999.99	1,594,999.99	675,000.00
Variable Costs	2,009,530.00	637,440.00	324,310.00	313,130.00	1,372,090.00	787,270.00	584,820.00
Allocated Fixed Costs	1,891,885.31	653,560.38	275,183.32	378,377.06	1,238,324.93	412,774.98	825,549.96
Net Profit	-641,715.33	-301,300.38	115,136.68	-416,437.06	-340,414.95	394,955.01	-735,369.96
Net Profit / Unit	-664.69	-200.60	40.12	-240.72	-464.09	443.77	-907.86

Picture 55: View on Products Profitability cube

Information which is offered by result of the allocation reveals that every sold Personal Account generates around 40 CZK, however offering Saving Account according to current conditions is highly unprofitable. Firstly, manager could see that revenue from Saving Account is much lower than in first one. Variable costs for both products are almost the same, although Saving Account consumes more than 100 000 CZK more in fixed costs than Personal Account. Similar situation is currently in Credit Accounts type products. At the first glance, it might look that Mortgages are profitable (difference between Revenue and Variable costs), however enormous amount of fixed costs which are consumed to offer Mortgages make it very bad output for bank to sell. Naturally, manager is able to observe other data from every component and trace root of the problem or anomaly. Some outputs of the allocation process as well as exemplary analysis he is able to execute will be presented in following paragraph introducing developed Excel based user interface.

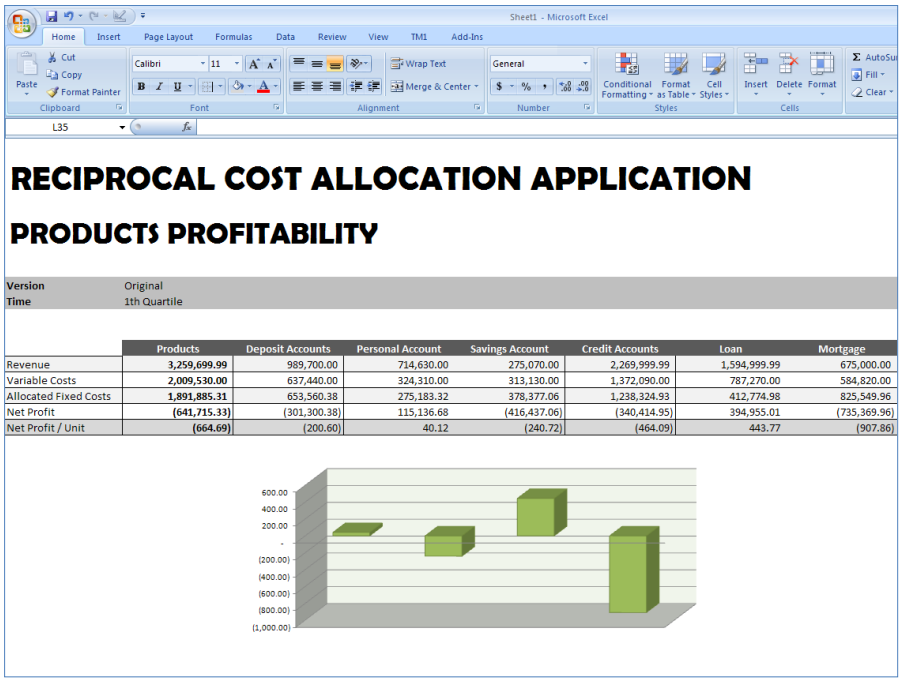
8.4. Development of User Interface

TM1 enables export of cubes including all functionality into MS Excel environment which offers leveraging of Excel advantages for business users in combination with efficient back end analytical system. Publishing cube and creation of connection between data on server and Excel cells is done simply by clicking on button in Cube Viewer. However this functionality is enabled only while working with Server Explorer running as add-in in MS Excel. Created spreadsheets could be further published on TM1 Web thick client and accessed via internet connection. In following screen shots I will describe two most important Excel sheets - Managers Console and Products Profitability. Considering the fact, that this kind of applications is primary dedicated for managers and business analytics, who are familiar with environment as well as capabilities of MS Excel, it will be appropriate to enable them managing whole application through this interface. Therefore I created introductory sheet which enable navigation and attendance of process with set of buttons divided into three main sections. As is depicted at picture no. 56, first column section called Allocation Drivers Setting enables user to access and edit driver's names in Drivers Names Console, values of particular drivers in Drivers Value Console and assign drivers to singular cost centers fixed costs in Drivers Assignment Cube. All this three buttons navigate user to sheets with exported corresponding cubes.



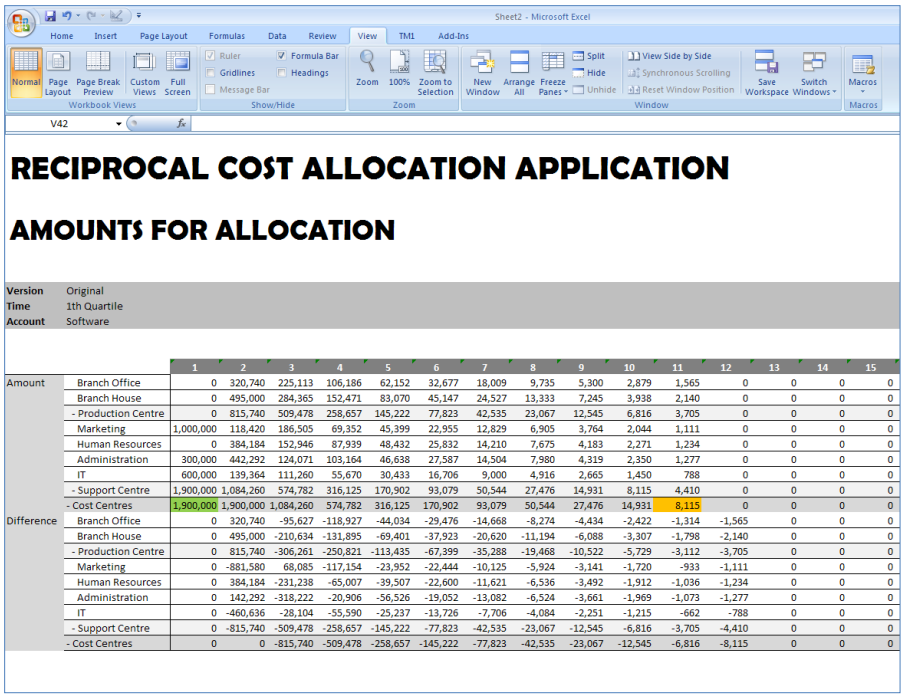
Picture 56: Excel based Managers Console

Second column of buttons provides different processes for loading dimension and data as well as navigate to cubes where what-if analyses could be executed. To Load dimension button are bounded all sub processes which we created for loading values as elements to dimensions such as *Products*, *Accounts*, *Time* and *Cost Center*. Furthermore we are able to load data from *Sales* table to *Sales&Production* cube via *Load SalesProduction* as well as fixed costs from *General Ledger* table *to Amounts for Allocation*. Last three buttons navigate user to sheets with published cubes where analyses could be executed. Final column of buttons is dedicated to allocation process itself. Besides the *Allocate!* Button which runs the major process, user can access *How to Allocate* cube and set method which determine number of iteration as well as corresponding value. After allocation process is executed, user is able to view results in *Products Profitability* cube and observe how where costs transferred in every step in *Multiplied* cube. Naturally, console could be adaptable and buttons could be organized as well as created further according to customer's requirements. In the second screen shot (picture no.57), we can observe Profitability cube exported and formatted in MS Excel. I leveraged functionality of Excel and graphically represented values from *Net Profit / Unit* for four leaf product elements in column chart. This offers better view on data, user is able to immediately identify that profitability of *Personal Account* is slightly above zero, second and fourth products are unprofitable and *Loans* generate highest net profit. As will be demonstrated further in exemplary analysis, chart is bounded with data on server and will be recalculated in a moment after change in values which affect *Net Profit / Unit*. Last screen shot *of Amounts for Allocation* cube in Excel shows, how was values allocated out and back on cost centers in particular steps of allocation. Green cells represents total amount of fixed costs which was required to be allocated. After 10 steps of allocation, we can observe that only 8 115 CZK remained unallocated (orange cells). We can check accuracy of the calculation by looking back to *Profitability* cube. Value in third row and first cell represents total amount of costs allocated on products. If we subtract 8 115 which remained unallocated from 1 900 000 which had to be allocated, result will be exactly 1 891 885 after rounding. 0, 31 difference is caused by various settings in number precious in both cubes. In amounts for allocation, I selected 0 precious.



	Products	Deposit Accounts	Personal Account	Savings Account	Credit Accounts	Loan	Mortgage
Revenue	**3,259,699.99**	989,700.00	714,630.00	275,070.00	2,269,999.99	1,594,999.99	675,000.00
Variable Costs	**2,009,530.00**	637,440.00	324,310.00	313,130.00	1,372,090.00	787,270.00	584,820.00
Allocated Fixed Costs	**1,891,885.31**	653,560.38	275,183.32	378,377.06	1,238,324.93	412,774.98	825,549.96
Net Profit	**(641,715.33)**	(301,300.38)	115,136.68	(416,437.06)	(340,414.95)	394,955.01	(735,369.96)
Net Profit / Unit	**(664.69)**	(200.60)	40.12	(240.72)	(464.09)	443.77	(907.86)

Picture 57: Products profitability cube formatted in MS Excel



RECIPROCAL COST ALLOCATION APPLICATION

AMOUNTS FOR ALLOCATION

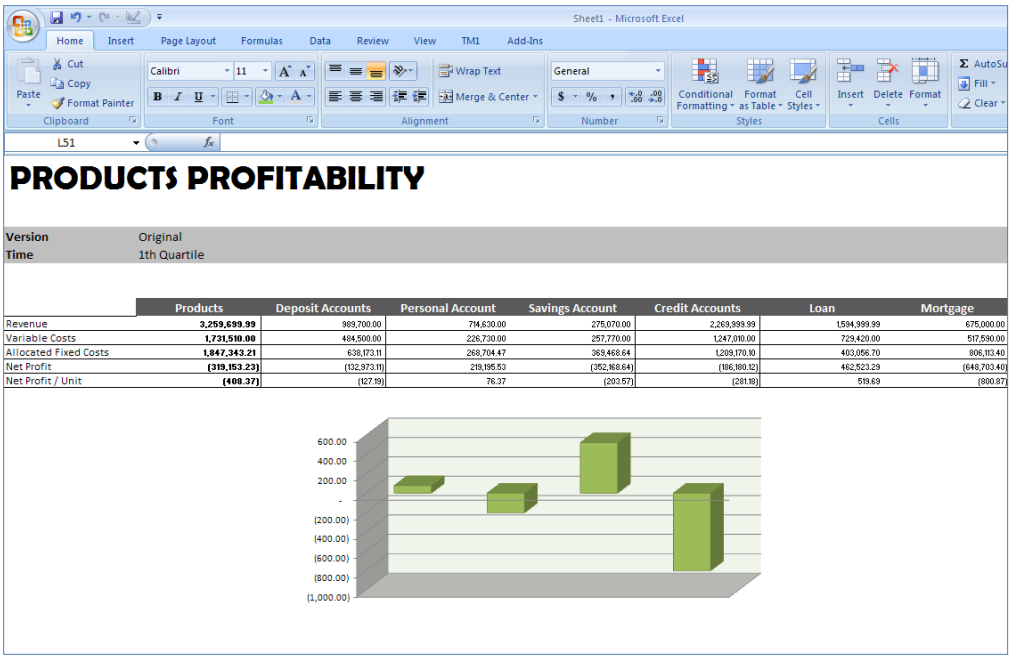
Version	Original
Time	1th Quartile
Account	Software

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Amount	Branch Office	0	320,740	225,113	106,186	62,152	32,677	18,009	9,735	5,300	2,879	1,565	0	0	0	0
	Branch House	0	495,000	284,365	152,471	83,070	45,147	24,527	13,333	7,245	3,938	2,140	0	0	0	0
	- Production Centre	0	815,740	509,478	258,657	145,222	77,823	42,535	23,067	12,545	6,816	3,705	0	0	0	0
	Marketing	1,000,000	118,420	186,505	69,352	45,399	22,955	12,829	6,905	3,764	2,044	1,111	0	0	0	0
	Human Resources	0	384,184	152,946	87,939	48,432	25,832	14,210	7,675	4,183	2,271	1,234	0	0	0	0
	Administration	300,000	442,292	124,071	103,164	46,638	27,587	14,504	7,980	4,319	2,350	1,277	0	0	0	0
	IT	600,000	139,364	111,260	55,670	30,433	16,706	9,000	4,916	2,665	1,450	788	0	0	0	0
	- Support Centre	1,900,000	1,084,260	574,782	316,125	170,902	93,079	50,544	27,476	14,931	8,115	4,410	0	0	0	0
	- Cost Centres	1,900,000	1,900,000	1,084,260	574,782	316,125	170,902	93,079	50,544	27,476	14,931	8,115	0	0	0	0
Difference	Branch Office	0	320,740	-95,627	-118,927	-44,034	-29,476	-14,668	-8,274	-4,434	-2,422	-1,314	-1,565	0	0	0
	Branch House	0	495,000	-210,634	-131,895	-69,401	-37,923	-20,620	-11,194	-6,088	-3,307	-1,798	-2,140	0	0	0
	- Production Centre	0	815,740	-306,261	-250,821	-113,435	-67,399	-35,288	-19,468	-10,522	-5,729	-3,112	-3,705	0	0	0
	Marketing	0	-881,580	68,085	-117,154	-23,952	-22,444	-10,125	-5,924	-3,141	-1,720	-933	-1,111	0	0	0
	Human Resources	0	384,184	-231,238	-65,007	-39,507	-22,600	-11,621	-6,536	-3,492	-1,912	-1,036	-1,234	0	0	0
	Administration	0	142,292	-318,222	-20,906	-56,526	-19,052	-13,082	-6,524	-3,661	-1,969	-1,073	-1,277	0	0	0
	IT	0	-460,636	-28,104	-55,590	-25,237	-13,726	-7,706	-4,084	-2,251	-1,215	-662	-788	0	0	0
	- Support Centre	0	-815,740	-509,478	-258,657	-145,222	-77,823	-42,535	-23,067	-12,545	-6,816	-3,705	-4,410	0	0	0
	- Cost Centres	0	0	-815,740	-509,478	-258,657	-145,222	-77,823	-42,535	-23,067	-12,545	-6,816	-8,115	0	0	0

Picture 58: Amounts for Allocation cube in MS Excel

8.5. Exemplary what-if analysis

Let's assume that manager is not satisfied with the current profitability of products and ask analyst to find some improvements. He is aware of board's consideration to purchase and implement ECM solution to solve paperwork, administration and compliance problems and would like to model the scenario of products profitability after successful implementation. From the managers console he navigates to amounts for allocation and insert value which will correspond to depreciation of software price in 1th quartile. It will be inserted into Software element in Account dimension and Administration in Cost Centrum. Further he has to consider impact of this change, which will also influence Average Paperwork per month driver. Anticipatory calculations estimate 90 percent decrease in rounding documents among cost centers as well as other time savings. He navigates therefore from manager's console to driver's values sheet and lower driver's values according to estimated percentages. He also does not have to forget to change variable's costs element *Paperwork* for every product. Last change will again involve fixed costs. Successful implementation might cause some personal changes in Administration department as well as save expenditures. Analyst has to estimate this value and consider the change. After editing all parameters, he could set the allocation method and execute allocation process right from manager's console. With values changed, output will look as follows



PRODUCTS PROFITABILITY

Version	Original
Time	1th Quartile

	Products	Deposit Accounts	Personal Account	Savings Account	Credit Accounts	Loan	Mortgage
Revenue	3,259,699.99	989,700.00	714,630.00	275,070.00	2,269,999.99	1,594,999.99	675,000.00
Variable Costs	1,731,510.00	484,500.00	226,730.00	257,770.00	1,247,010.00	729,420.00	517,590.00
Allocated Fixed Costs	1,847,343.21	638,173.11	268,704.47	369,468.64	1,209,170.10	403,056.70	806,113.40
Net Profit	(319,153.23)	(132,973.11)	219,195.53	(352,168.64)	(186,180.12)	462,523.29	(648,703.40)
Net Profit / Unit	(408.37)	(127.19)	76.37	(203.57)	(281.18)	519.69	(800.87)

Picture 59: Result of exemplary what-id analysis presented by chart

Profitability of product will be immediately recalculated as well as chart will be rebuilt. Movements of the columns are not very well visible, however if we check *Net Profit* of all products, scenario is considered as effectual improvement.

IV. Findings

Writing thesis of wide comprehension and detail coverage of theoretical as well as practical "World of Performance Management" I contemplated to offer usual reader complex understanding of problematic and primarily map it to current financial and fiscal situation so that he senses the necessity of exploiting and implementing solution of similar capabilities. For people working in field or with technology I intended to make thesis propositional especially in Case Study, where some advanced functionality was explained as well as projection of implementation was explained in detail. Third group of users for who was thesis dedicated are CFOs and financial managers. Thesis might propose them appropriate solution to support their decision making and improve performance through scenario simulations and analyses capabilities.

The main contribution of thesis I see in linking together Managerial Accounting and Business Intelligence. Execution Information Systems are primary dedicated to business user, who understands implemented problematic. One of the key requirements before implementing is therefore placed on separation from "hands" of IT and administration of business department by itself. This markedly truncates business requirements chain and their translation into IT language to solely direct business queries. However administration of business application demands also its development by business users. TM1 is considered to be appropriate software, where applications are possible to by develop by users familiar with MS Excel. My aim was therefore also to demonstrate how simply could be whole application developed and how flexible is its administration as well as modification.

Significant part of the thesis was dedicated to Case study. As beneficial I also consider reasoning how to approach to project of this kind where I explained and describe every part of solution from data source analysis to particular implementation. In architecture and design section of the solution, which I consider as the most important part of the whole project, I tried to demonstrate how to decompose complex problematic to singular units, in our case represented by cubes and dimensions and how to organize them to meet all requirements and to satisfy condition for flexible modification according to possible demands.

My selection of the topic was strictly influenced by current economical situation as well as possibility to link together Managerial Accounting and Business Intelligence which I am concerned about. I will be very pleased to elaborate my experience with Performance Management in the future as well as to contribute to its field with some innovative and helpful solutions.

I do not see better way how to conclude thesis than use one of the most important cogitation from the beginning and along this line makes from it one complex entity.

"Indications, answers, right solutions. They are all around us. The only we need is the proper way to collect, store and harvest true value from them."

As a consequence I hope that thesis has accomplished the goal I appointed at the beginning and drew on the concept of taking advantage of one of the greatest worth we have – information and mainly specified ways how to "harvest" this true value from them.

V. Dictionary

Term	Acronym	Explanation [source]
Business Intelligence	BI	Business Intelligence is set of processes, applications and technologies which goal is to effectively and advisably support decision making in the company. They assist in analytical and planning activities and are built on multidimensional principles of observing company's data. [freely translated from Novotný O., Pour J., Slánský D., 2005, p. 19]
Corporate Performance Management	CPM	Corporate (Business, Enterprise) Performance Management is a key business initiative that enables companies to align strategic and operational objectives with business activities in order to fully manage performance through better informed decision making and action. [Ballard Ch., White C., McDonald S., 2005, p. 27]
Data Warehouse	DW	Data Warehouse is integrated, subjectively oriented, permanent and timely diversified set of data, aligned to support management requirements. [freely translated from Novotný O., Pour J., Slánský D., 2005, p. 32]
Enterprise Content Management	ECM	Enterprise Content Management refers to the technologies, strategies, methods and tools used to capture, manage, store, preserve, and deliver content and documents related to an organization and its processes. ECM tools allow the management of an enterprise level organization's information. [WWW source, Enterprise Content Management]
Key Performance Indicator	KPI	KPI is a performance metric for a specific business activity that has an associated business goal or threshold. The goal or threshold is used to determine whether or not the business activity is performing within accepted limits.[Ballard Ch., 2008, p. 32]
On line analytical processing	OLAP	OLAP is information technology based on concept of multidimensional databases. Main principle of OLAP is multidimensional table enables quickly and flexibly manipulates its dimensions and allows user changing the view on the modeled economical situation. [freely translated from Novotný O., Pour J., Slánský D., 2005, p. 21]

VI. Literature

WWW sources:

Enterprise Content Management, available at WWW:
http://www.aiim.org/What-is-ECM-Enterprise-Content-Management.aspx

Ganesh B., Edwards R., Burnett S., Jennings T.: Business Intelligence (Corporate Performance Management), Butler Group Research, 2009, available at WWW:
http://www.butlergroup.com/research/research.asp

IBM Smarter Planet – New Intelligence, available at WWW:
http://www.ibm.com/smarterplanet/us/en/business_analytics/ideas/

TM1 Reference Guide 9.4.1, available at WWW:
http://publib.boulder.ibm.com/infocenter/ctm1/v9r4m1/index.jsp?topic=/com.ibm.swg.im.cognos.tm1_ref.9.4.1.doc/tm1_ref.html

Wikipedia – Cost Drivers, available at WWW:
http://en.wikipedia.org/wiki/Cost_driver

Publications:

Ballard Ch., Farrell D., Gupta A., Mazuela C., Vohnik S.: Dimensional Modeling: In a Business Intelligence Environment, An IBM Redbooks publication, 19 May 2008, ISBN 0738496448, 664 pages, available at WWW: < www.redbooks.ibm.com >

Ballard Ch., White C., McDonald S., Myllymaki J., McDowell S., Goerlich O., Neroda A.: Business Performance Management . . . Meets Business Intelligence, An IBM Redbooks publication, June 2005, ISBN 0738493635, 224 pages, available at WWW: < www.redbooks.ibm.com >

Bothe O., Pavlík M.: Types of costs allocation, Internal materials of IBM Corp.

Fibrílová J., Šoljaková L., Wagner J.: Nákladové a manažérské účetníctví, 2007, Praha ASPI, ISBN 978-80-7357-299-0, 432 pages

Hnilica J., Fotr J.: Aplikovaná analýza rizika ve finančním managementu a investičním rozhodování, Grada Publishing, 2009, ISBN 978-80-247-.2560-4, 264 pages

Kaplan R., Cooper R.: Cost and Effect: Using Integrated Cost Systems to Drive Profitability and Performance, Harvard Business School Press, 1998, ISBN 0875847889

Novotný O., Pour J., Slánský D.: Business Intelligence, 2005, Grada Publishing, ISBN 80-247-1094-3, 256 pages

Popesko B.: Moderní metody řízení nákladů, 2009, Grada publishing, ISBN 978-80-247-2974-9

Pour J., Gála L., Šedivá Z.: Podniková informatika 2. přepracované a aktualizované vydání, 2009, Grada Publishing, ISBN 978-80-247-2615-1, 496 pages