

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

System Engineering and Informatics



Diploma Thesis

Quality Metrics for Business Process Modelling

Author: Adil Sylqa

Supervisor: Ing. Josef Pavlíček Ph.D.

© 2016 Prague

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

DIPLOMA THESIS ASSIGNMENT

Adil Sylqa

Informatics

Thesis title

Quality Metrics for Business Process Modelling

Objectives of thesis

The main goal is to identify current state of business process models design. Try to identify fundamental metrics usable for calculation quality of design and which from found metrics can be use to prediction of design quality. Other goal is to identify, how found metric can influence the design quality during the process modeling.

Methodology

Analyze current state of business process modeling.

Study suitable literature for searching current state of process models metrics.

From the result of study define current metrics state.

Recommend the metrics, which can influence quality of business models.

Try to preset the study results in the scientific journal.

The proposed extent of the thesis

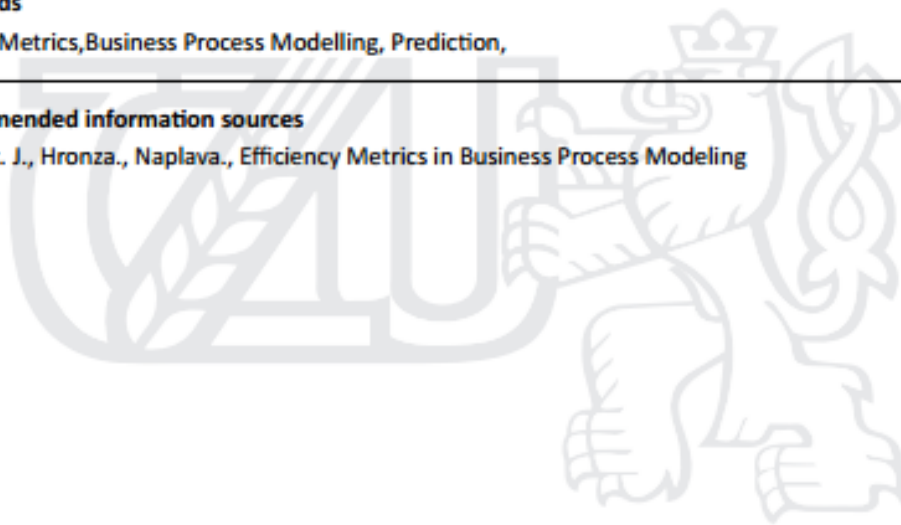
70 pgs

Keywords

Quality Metrics, Business Process Modelling, Prediction,

Recommended information sources

Pavlicek. J., Hronza., Naplava., Efficiency Metrics in Business Process Modeling



Expected date of thesis defence

2015/16 SS – FEM

The Diploma Thesis Supervisor

Ing. Josef Pavlíček, Ph.D.

Supervising department

Department of Information Engineering

Electronic approval: 4. 3. 2016

Ing. Martin Pelikán, Ph.D.

Head of department

Electronic approval: 4. 3. 2016

Ing. Martin Pelikán, Ph.D.

Dean

Prague on 20. 03. 2016

Declaration

I hereby declare that I have worked on this diploma thesis, named “Quality Metrics for Business Process Modelling”, myself and I have used only the sources listed in this paper.

In Prague,

.....

Adil Sylqa

Acknowledgment

All thanks due to God for providing me with the will and energy to work on this diploma thesis. My sincere gratitude goes to Ing. Josef Pavlíček Ph.D. for trusting me to tackle this subject, his constructive advices were excellent and I have benefited immensely from having him as a supervisor.

Many thanks go to my family and friends for supporting me unconditionally throughout all my studies.

Quality Metrics for Business Process Modelling

Míry Kvality Procesních Modelů

Summary

In the field of Business Informatics today more and more importance is given to business process models since it serves as a foundation for communication between the stakeholders in the software development process. In the field of software engineering, quality metrics have proven to be an industry changer in terms of good programming practices and software design. This paper shows similarities found between software applications and business process designs and stresses out the potential of quality metrics in BPM.

To have a competitive edge in the process oriented approach it is necessary for these models to have very good quality metrics so we can measure and create easy to understand, unambiguous and less error prone models. The current existing metrics although measure different complexity values within a model, there is not a clear standard that captures process clarity and design quality. In the practical part of this thesis we define a new potential metric (Compound Complexity) that can grasp and measure in a more rounded way the different quality aspects of a business process model through the use of BPMN notation.

The case study presented in this paper, shows that Compound Complexity when presented with two almost identical models, can differentiate between the less and the more complex one. The results show there are measurements that can describe a large portion of process clarity and understanding in a quantifiable metric, however there is still more empirical research that needs to be done in order to validate Compound Complexity as a sophisticated quality metric.

Keywords

Business Process Modelling, BPM, quality metrics, BPMN, business process, management, software development

Shrnutí

Práce se zabývá problematikou výpočtu měř složitosti procesních modelů modelovaných dle BPM notaci. Rozpoznat a bezpečně měřit složitost procesních modelů je velmi účelné v aplikované informatice, neboť procesní modely jsou základními stavebními kameny návrhu především rozhodovacích - manažerských informačních systémů. Zatímco metody uložení dat (uvažujme typicky SŘBD), metody kvalitního strukturálního (struktogramy) či objektového (UML – diagramy tříd) návrhu jsou propracovány velmi podrobně, v oblasti procesního modelování vhodné míry a postupy stále chybí. Práce se inspiruje propracovanými technikami v oblasti programování a na základě analýzy nabytých znalostí se autor pokouší navrhnout nové míry složitosti určené pro BPM notaci. Tyto míry jsou rozebrány v příkladové studii a na porovnání jednotlivých modelů (neboť jeden proces je možné navrhnout několika způsoby) autor prokazuje možnost zavedení nové míry složitosti. Autor závěrem uvádí použitelnost nové míry a navrhuje směry, kterými by se měl ubírat další výzkum zvolené problematiky.

Klíčová slova

Business Process Modelling, BPM, quality metrics, BPMN, business process, management, software development

Contents

1.	Introduction	5
2.	Objectives & Methodology	7
2.1.	Objectives	7
2.2.	Methodology.....	8
	Literature Review	9
3.	Introduction to the Process Oriented Approach	9
3.1.	Business Processes	9
3.2.	Business Process Management.....	11
3.3.	Business Process Reengineering	12
3.4.	Business Process Innovation	13
4.	Business Process Modelling.....	16
4.1.	Overview	16
4.2.	Business Process Modelling Techniques.....	18
4.3.	Business Process Model and Notation (BPMN)	19
4.4.	Basic Concepts of BPMN.....	20
4.5.	Designing a Business Diagram.....	24
5.	Quality Metrics for BPM – Current State	29
5.1.	Control-Flow Complexity (CFC)	30
5.2.	Number of Activities (NOA).....	32
5.3.	Coefficient of Network Complexity (CNC).....	33
	Practical Part	36
6.	Metrics Proposal - Feature Diagrams Compound Complexity	36
6.1.	Feature Diagrams in Domain Models.....	37
6.2.	Complexity Measures in Feature Diagrams	38

6.3.	Calculating Metrics for a Feature Diagram	42
7.	Applying Feature Diagrams Compound Complexity in BPM.....	46
7.1.	Setting Cognitive Weights for BPMN Elements.....	47
7.2.	Metric Calculation	51
8.	Metric Testing and Comparison – Case Study.....	60
8.1.	FactOrEasy Game Overview	60
8.2.	FactOrEasy Model Design	62
8.3.	Model Evaluation by Compound Complexity.....	67
8.4.	Model Evaluation by Existing Quality Metrics.....	68
8.5.	Evaluation of Results.....	71
9.	Conclusion and Future Work	72
10.	References	74
11.	Table of Figures	78
12.	Appendix A – Sub-processes for Models 1 & 2	80

1. Introduction

With the exponential growth of technology in the past decades, Information Technology has found itself intertwined with almost every organization. As a consequence a lot of organizations are choosing to be process oriented. To be process oriented means focused on business practices that differ from customer to customer rather than putting emphasis on the traditional hierarchical structures (1). This allows organizations to be more flexible and develop an edge when dealing with a variety of customers. They have realized the importance technology plays in improving their efficiency and quality of their business processes through the use of Business Process Management. A lot of money is being spent in the IT resources in order to facilitate better models for business processes in their day to day use.

However, even though business process models are enjoying quite an extensive use among different industries, there is a lack of an empirical focus on tracking quality and usage aspects of process modelling (2). The demand for BPM use will only get bigger in the future so this is why important to develop some metrics that will be able to tell us in a clearer picture if organizations are really capitalizing in this process oriented approach, what are the bottlenecks, which aspects of business processes should have priority and ultimately provide necessary feedback on what needs to be improved in the model.

It is fairly common for medium to large organizations to build their own repositories of BPMs that would serve as a knowledge base for their business process management efforts. These repositories can reach thousands of business process models (3). In this paper we will use as a reference, a research done in Czech Technical University that deals with improving the academic institution through business process modelling. In this project there were described more than 400 business processes in Business Process Modelling Notation (BPMN). There were proven to be a various issues with the implementation of the models but the main bottlenecks were:

- Inconsistent level of detail as a consequence of no standardized procedures used by different individual creators
- Change of participant's role during the execution of business process

- A big number of BPMN symbols within a business process led to confusion
- Multiple and unnecessary use of same BPMN symbols
- Different levels of distribution of one business process into multiple sub-processes

What should have been easy to understand, simple models, if not properly evaluated the can often led to the redesign of the whole business process which not only wastes resources but also doesn't guarantee reusability in the future.

Through literature review, theoretical and practical perspectives this paper aims to define if business process models can be measured, how can they be measured and is it possible to use a standard when measuring business process models.

The remainder of this paper is structured as follows. The next section will present the objective and methodology used, followed by an overview of business process modelling, current state of BPM quality metrics, research findings, interpretation of results and finally the last section provides conclusion with directions for future research.

2. Objectives & Methodology

2.1.Objectives

. Similarly a business process consists of activities and operations which resembles a traditional software program.

There has been a lot of research on the complexity metrics field in software development that has had an immense influence in software re-engineering. The aim of this thesis is to analyse, adapt and quantify these complexity metrics for measuring business practices and from the recent work done in complexity measure try to propose ideas for new metrics which then would be used by process analysts to design proper models and incite business process re-engineering.

Partial goals of this thesis are:

Literature overview on the topic

A definition of business process modelling on an introductory level

Analyse and implement new metrics that can improve the quality measurement

Evaluate and test the new metric in a BPM of a real-life application

Compare results with existing metrics

2.2.Methodology

Methodology used in this thesis is based on an in-depth overview of relevant literature in order to identify the current state of business process design and how quality metrics are able to influence better designs.

Practical approach aims to analyze the current state of process model metrics from different approaches of complexity measurement in many similar fields to BPM like software engineering, product line engineering and graph theory, we will try to define and adjust new quality metric that is able to capture design quality and influence quality of business process models.

In this thesis, we try to break down the idea behind this set of metrics based on an empirical approach stemming from software development practices and in the process propose new measurements in the form of metrics. For proposing new metrics, a study on the most recent papers that discuss complexity measures in similar fields will be conducted.

Regarding metric evaluation and testing, we will design a variety of process models for a real-life application using BPMN notation (Bizagi Modeller will be our software of choice) and those models will be submitted for evaluation from our proposed metric.

As part or result evaluation there will be a comparison between the newly proposed metric and the existing metrics which will tell us the state of our developed metric in measuring different aspects of quality in a business process model. There will be an attempt to present these results in a scientific journal

Literature Review

3. Introduction to the Process Oriented Approach

The following chapter tries to describe briefly the definition of business processes, lay down the core features of the process oriented approach and describe how IT has been an instrumental factor on emphasizing processes as key to an organization's growth.

3.1. Business Processes

Before we move forward and assess the modelling or management of business processes it is essential to have a firm understanding on what is a business process. The term itself can be very ambiguous and lead to many interpretations so a clear definition is needed. Business Process can be defined as “a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer” (4) or “a specific ordering of activities across time and place, with a beginning and an end with clearly defined inputs and outputs” (5).

A process basically consists of a set of attributes distributed into a flow of steps with the end-goal being to achieve a particular task. Generally, processes help in managing the operations of an organization so it can produce valuable outputs.

According to the second definition we can divide business processes within an organization into *operational* and *management* processes. In operational processes, the execution of tasks comprises the activities of an organization's value chain while management processes are associated with the administration, control and distribution of resources within organizations.

It is important to differentiate between processes that are associated with business operations and those processes that associate with information handling, coordination and control in order to be able to ensure the efficiency and success of the primary business operations.

A well-organized flow of both kinds of processes acts as a foundation not only for the reliability of an organization but also is a catalyst for improving efficiency. A process view approach can determine extra factors affecting the conversion of IT assets to a positive or negative impact (6). To illustrate this, let's say a company decides to use some kind of IT assets to improve their market value and gain investments as an end goal. If they succeed and the use of IT assets proves to have had a direct influence on reaching this end goal then the business value of IT is recognized as a positive impact. Hence, it is said that analysing business processes of an organization will allow for better results when analysing the impact of IT in an organization by identifying the mechanisms that add value and purge those that don't. (7)

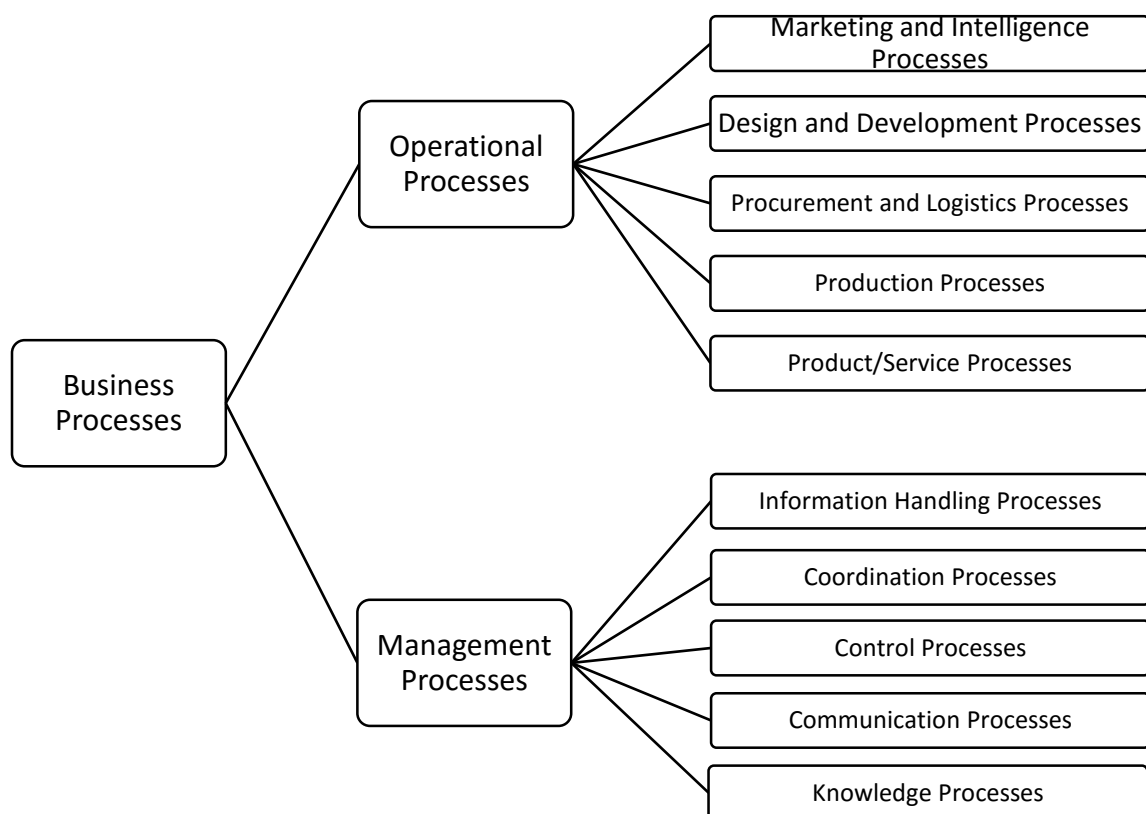


Figure 1 - Typology of Business Processes¹

¹ Mooney, John; Gurbaxani, Vijay; and Kraemer, Kenneth, "A Process Oriented Framework for Assessing the Business Value of Information Technology" (1995). ICIS 1995 Proceedings. Paper 3

The main reason business processes are maintaining organizations edge ahead of their competition is attributed to the rapid evolution of Information Technology. To back up this argument one research shows that about 46% of all equipment spending in USA is being spent on improving the organization efficiency with IT equipment and software (8). This penetration of software in the business sector only means that organizations are now able to constantly improve efficiency by redesigning their business processes so they can attract more market share in a very competitive environment. To be able to better understand business process modelling it is essential to briefly discuss about Business Process Management (BPM), Business Process Re-Engineering (BPR) and Business Process Innovation (BPI).

3.2. Business Process Management

Business Process Management theoretically can be defined as “a management discipline focused on using business processes as a main contributor to achieve objective through the improvement and governance of essential business processes” (9). In practice, BPM has been used in much more broader concept. The history of BPM goes back to 1990s where it was supposed to be the natural successor of the Workflow Management. The main reason for this evolution lied in the changes the business environment was experiencing. In the past Workflow Management (WFM) was an information system that was managing its schemas (i.e. control flows) through the use of software with a fundamental assumption that workflows should be predefined.

With the rapid increase of World Wide Web, the predefined logic of WFM was considered a weakness and in order to adapt the new business environments which were changing from centralized-and-closed to distributed-and-open, there was a demand for a new way of managing processes (10). These were the early stages where BPM started making an impact as a more holistic approach to organizing work across all resources (including workflows). This initiated the departure of workflows being interactions between people and software systems to a universal language that is used in the same manner from people and software systems.

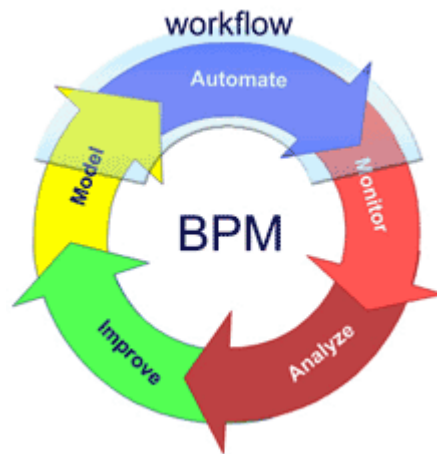


Figure 2 - The evolution of BPM [source: <http://dashflows.com/workflows>]

In recent times BPM has outgrown itself and it converges concepts like Workflow Management (WFM), Case handling (CH), Enterprise Application Integration (EAI), Enterprise Resource Planning (ERP), Customer Relation Management (CRM) etc. (11)

3.3. Business Process Reengineering

The person who introduced the concept in the business environment was Michael Hammer in the early 90s. His classic definition for Business Process Reengineering (BPR) is defined as “the fundamental rethinking and radical redesign of business processes to achieve dramatic improvements in critical, contemporary measures of performance, such as cost, quality service, and speed” (12). The essence of BPR is to make a gradual enterprise revolution, and its main achievement aims to redesign and restructure the process which has been divided into fragments by specialized division and the bureaucratic system (13).

BPR aims to break from current processes by suggesting new ways of organizing tasks, people, and it emphasizes the role of IT systems in improving processes to better support the goals of the organisation. This is done by identifying, analysing and finally

redesigning business processes in such a manner that each redesigned process will be more efficient than the previous one.

According to Vidgen et al. (1994) the central principles of BPR are:

- Radical change and assumption challenge
- Process and Goal Orientation
- Organisational Re-Structuring
- Exploitation of current technologies, mainly Information Technology (14)

The reengineering theory has proven to be applicable in today's ever expanding business environment. Researches have shown that technology has been incorporated to support the reinventions of business processes (15). Organizations are dealing with a lot of challenges adapting their business services in the fast pace nature of changes in technology, regulations and stiff competitors. With so many competitors in the market, organizations are trying to maintain their edge by improving the quality of services with short turnaround time.

This change in approach has made obsolete traditional management mechanisms as a mean of satisfying their clients. On the contrary, relying only on IT to meet objectives in the new era hasn't proven feasible, so organizations need to look into their core processes to make the necessary changes. Hence, by using BPR, organizations are able to redesign their processes in such a way that they are not dependent on the current IT trends since software here is used as a tool that helps automate processes and not something that can determine them.

3.4. Business Process Innovation

According to Schumpeter (16) classic definition, an innovation in business can be (i) a new way of handling processes, (ii) a new product, (iii) a new target market, (iv) new supply sources, and (v) new competitive structure in an organization. A more general approach by Rogers defines innovation as any idea, practice or object that is perceived to be new by an individual or other units of adoption (17).

Another distinction on types of innovation is described by Damanpour who separates them into two types: *technical innovations* and *administrative innovations*. Technical innovations deal with implementing new processes, products or services while administrative innovations deal with implementing new procedures, policies and organizational forms (18).

A necessity of the modern IT market is the ability to innovate. With the recent success of social media services like Facebook, Twitter, Uber who started as small startups and now are among top earning organizations, more and more money are being thrown into startup culture as a way of innovating new products or services. Although innovation can be hard to achieve in a competitive market, researches show that organizations are looking to use the power of Internet and software to achieve process innovation and process reengineering (19).

It is clear that BPR and Business Process Innovation (BPI) are linked together since BPI creates a more effective way of handling a process which then initiates process reengineering. A key element in this BPR-BPI connection is the IT factor since it is the main facilitator of innovation processes. With powerful tools like ERP and CRM, backed by cheap networks, companies are swiftly replicating Business Process Innovation throughout their organizations (20).

In a recent research done in process innovation (20), McAfee and Brynjolfsson suggest three main points that organizations should follow:

- 1) **Deploy** a consistent technology platform rather than pasting together a variety of legacy systems
- 2) **Innovate** better ways of working. Best practices for innovation are processes that:
 - a) Apply consistently in most, if not all, departments of the organization (such as stores, factories, delivery teams)
 - b) Give preliminary results and metrics as soon as the IT system starts being used
 - c) Require precise instructions (such as order taking or delivery)
 - d) Can be executed the same way everywhere and every time in the organization
 - e) Can be tracked in real time so in case of severe issues, a rollback can be applied immediately

- 3) **Propagate** these process innovations widely throughout the organization by using IT.

Deploying enterprise IT can be a challenge for most companies because it is a big financial investment so it requires careful configuring and testing making sure it is fitting to the needs of company.

On top of that there is the behavioral change of the employees who view enterprise IT as something that will change their jobs in a major way but with a ruthless competitive environment the only way to survive is to innovate and as data shows, innovation comes from a process oriented approach that is enabled by the effective use of IT.

4. Business Process Modelling

4.1. Overview

In today's world, information has become an indispensable asset which has to be carefully planned, organized and documented. Just like information, business processes need to have similar treatment and there are empirical arguments that link organization's potential for improvement with the optimization of their business processes. The optimization of business processes happens when there is a clear view of all the information structures which in turn makes it easy for business process to be analyzed and redesigned. This defines the role of the business process modelling. (21)

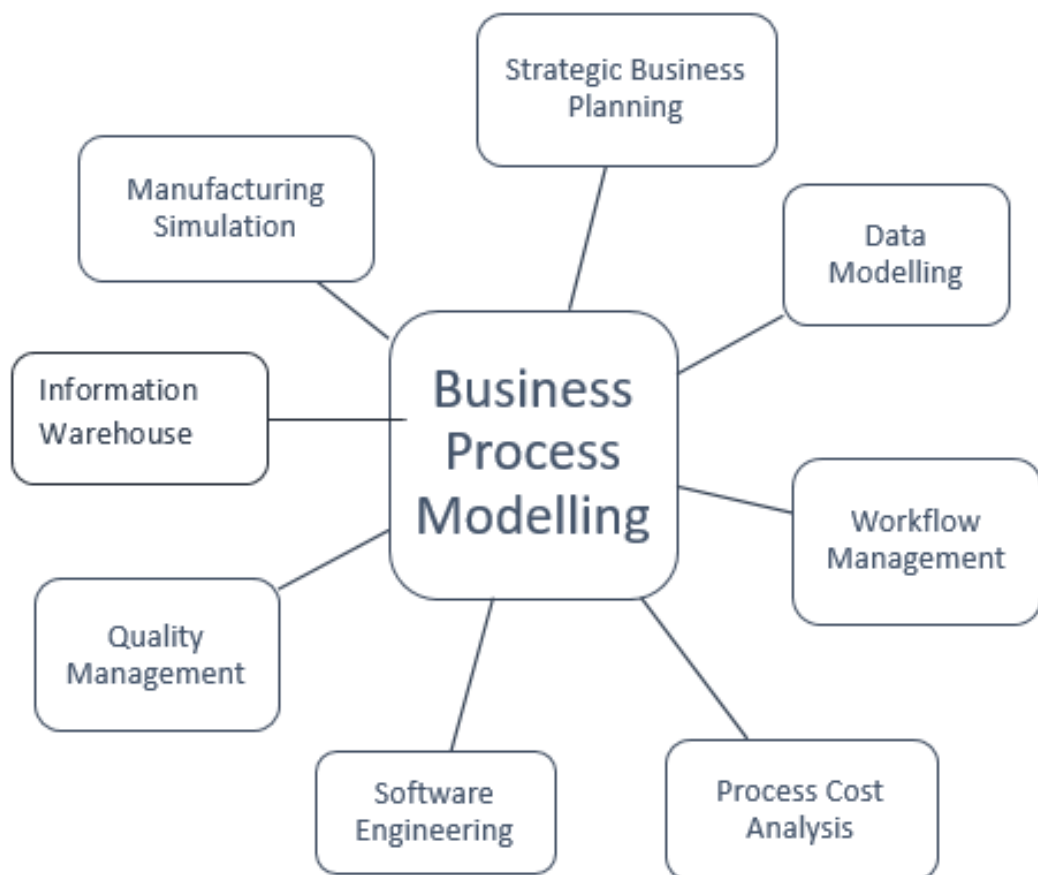


Figure 3 - Interaction Areas of Business Process Modelling [author]

By using different kind of modelling visualizations, BPM is used to present information about a process and describe the interaction within or between organizations involved either in creating this process or implementing it. It can show multiple granularity levels beginning from a depiction of a simple workflow to simulation and execution. Business Process Models are also very good in addressing complexity levels inside a process by emphasizing on specific aspects and reusing those models. (22)

Business Process Modelling is cross-platform, it takes into consideration the processes that are not bounded to one department but to the whole organization. It also can include external activities if they fit into the system of primary processes.

In order to develop a sustainable BP model, project members need to understand well current and desired business processes so they can make informed decisions in compiling the requirements needed to either optimize current processes or develop new ones.

Usually the cost of developing new processes from the beginning is very high so it is very important to focus on the factors that ensure quality of the model before the processes are modelled. It is impossible for organizations to know exactly how their processes will grow in the long-term because the nature of business is as dynamic as ever so that is why it is important to take a scalable approach and first focus to develop core business processes that are not dependent on the emerging technologies.

After the core business processes have been defined and documented, the organization can gradually build on them and develop more advanced models.

There are many factors that determine good practices when creating business process models and those will be discussed in more detail in the practical part of this paper but some of the points that should be considered before spending resources on Business Process Modelling are:

- The scope of the model
- Participation of all the members
- Making sure the modelling gains outweigh its costs
- Business Process Models have to be understandable by everyone
- Participants have to understand how BPM benefits their work

When there is enough level of certainty among the above mentioned points and organization decides to model their business processes, BPM should iterate between these 4 phases:

1. Review requirements needed to reach the business goal
2. Identify users, processes and participants
3. Use one or some of the BPM techniques to create a diagram
4. Review business process analysis

One of the goals of BPM is also to achieve an understanding of business knowledge between the executives and IT engineers so it drives the design and implementation of software systems tailored to organization's needs.

4.2. Business Process Modelling Techniques

There are many ways that can visualize a business process model. It can be through a modelling language, notation, framework or within an overall enterprise architecture. Most of them share the same idea behind it which is to map a process so people can understand, analyze and bring changes to that process. This is usually done by drawing a sort of diagram.

Designing a business process model through a modelling language and notation like UML or BPMN (Business Process Modelling Notation) is sufficient to model complex processes while being easily understandable for less tech savvy people in the organization.

In the other hand a modelling framework covers the whole lifecycle from requirements analysis to disposal phase. Modelling frameworks are good for model's reusability, be that as a whole or some part of it.

Some of the different techniques that are used for business process modelling are:

- Business Process Modelling Notation (BPMN)
- UML Diagrams
- Flowchart
- Data flow diagrams

- Role activity diagrams
- GANTT charts
- Coloured Petri Nets (CPN)
- Object oriented methods
- Workflow
- Simulation model

For the purpose of this paper, we will use BPMN to describe the visualization of the models because it is one of the most used methods to apply modelling and it is really easy to understand for both IT experts and non-technical people.

4.3. Business Process Model and Notation (BPMN)

Business Process Model and Notation (BPMN) is a graphical notation that describes the logic of steps in a business process. It provides a common language which allows all involved members to communicate processes clearly, completely and efficiently (23). It was first defined by the Business Process Management Initiative (BPMI) in 2004 and then in 2006 adopted by Object Management Group (OMG). Version 2.0 of BPMN was released in January 2011 (24).

There are many reasons why is it important to model with BPMN.

- a) BPMN is a worldwide process modelling standard,
- b) it is independent from any modelling methodology,
- c) it bridges the gap between business processes and their implementation and
- d) it enables modelers to create processes in a unified way so everyone can read the models with a common understanding.

BPMN provides many advanced options to describe concepts like transactions, compensations and exception handling. These features make BPMN well suited for generating executable processes for languages like BPEL (Business Process Execution Language). However with the release of BPMN 2.0, the execution semantics were introduced alongside the notational and diagramming elements (24).

In this way by using the BPMN semantics in a Business Process Diagram (BPD), diagrams become very easy to understand. BPD shows a graphical sequence of all the activities and events that take place during a process. It also contains additional information that is helpful for making an analysis.

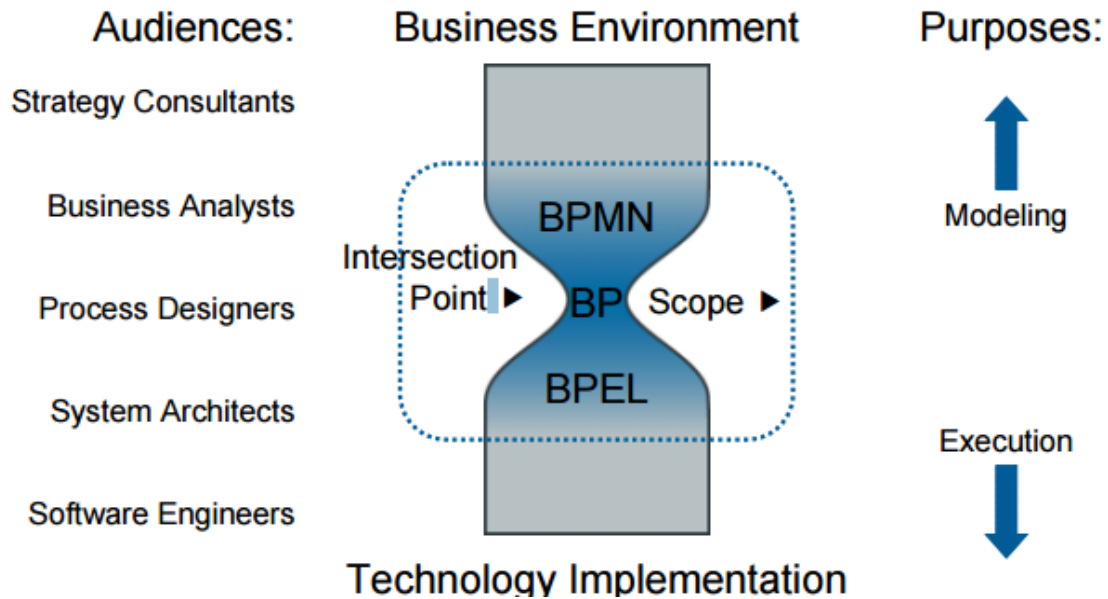


Figure 4 - A BPM Hourglass²

4.4. Basic Concepts of BPMN

Before we are able to design business process models it is important to explain the basic diagram concepts that are used for modelling.

Flow Objects – are the main elements that define processes behavior. They consist of:

- *Events*

An event is something that “happens” during the course of a business process. They can start, interrupt or end a flow and usually initiate a trigger or a result. Events are denoted by circles and the type of boundary determines the type of event.

² Source: Stephen A. White, Introduction to BPMN, IBM Software Group



Figure 5 - Events [author]³

- **Activities**

Activities represent the work that is performed inside a business process. They are denoted by rounded rectangles. The activities can be compound or simple, so they are divided into **Tasks** and **Sub-Processes**

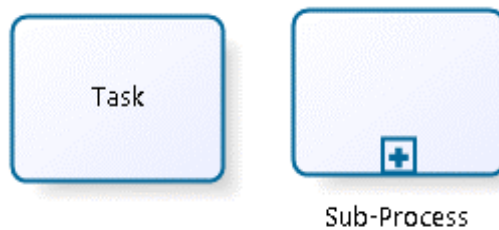


Figure 6 - Activities [author]

There are different kind of Tasks (Simple, Automatic, Manual, etc.) as well as different Sub-Processes (embedded, reusable, etc.) which allow for modelling in greater depth and in the same time offer more clarity for the reader.

- **Gateways**

Gateways are modelling elements that are used to control the convergence and divergence of the flow if it is necessary. If the flow does not to be controlled then gateways are not needed. They are denoted by diamonds. There are 5 main gateway types

³ All BPMN objects used by author are designed Bizagi Modeler freeware



Figure 7 - Gateways [author]

Usually the most used type of gateways are Exclusive Gateways or else known as XOR Gateways. This is present when process flow is in a crossroad and only one of the outgoing paths can be taken when the process is executed. The other gateway type is Inclusive, where there is more than one possible outgoing path. Parallel Gateways are used in processes that have multiple parallel paths defined and Complex Gateways are decisions that take into consideration more advanced definitions of process behaviour.

Connectors – are elements that connect two different objects in the process flow. There are 3 types of connectors: *Sequence Flows*, *Associations* and *Message Flows*

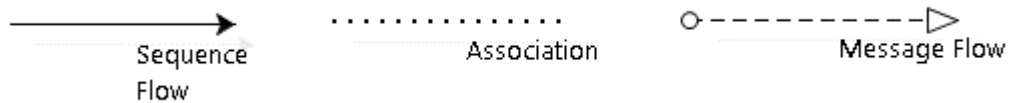


Figure 8 - Connectors [author]

Sequence Flow is the most basic connector. It is used to show the order that activities will be executed in the process. Each flow must have only one source and one target. *Association* is used to associate information and Artifacts with Flow Objects. *Message Flow* is used to show the flow of messages between two entities that are prepared to send and receive them.

Swimlanes – are a BPMN concept that is used to partition and organize the process flow by a visual hierarchy which can represent functional areas, roles or responsibilities.



Figure 9 - Swimlanes [author]

A *Pool* represents a participant in the process. A participant can be a business entity (company) or a general business role (buyer, seller, manufacturer etc). A *Lane* is a sub-partition inside a pool and is mostly used to further distinct roles inside a business entity. A *Milestone* is just a sub-partition inside the process which is used to track milestones.

Artifacts – are elements that provide additional information beyond the basic sequence flow structure of the process. There are usually three types of artifacts used in BPMN: *Data Objects*, *Groups* and *Annotations*. However, based on certain modellers there can be new types of Artifacts like images, headers or user customized artifacts.

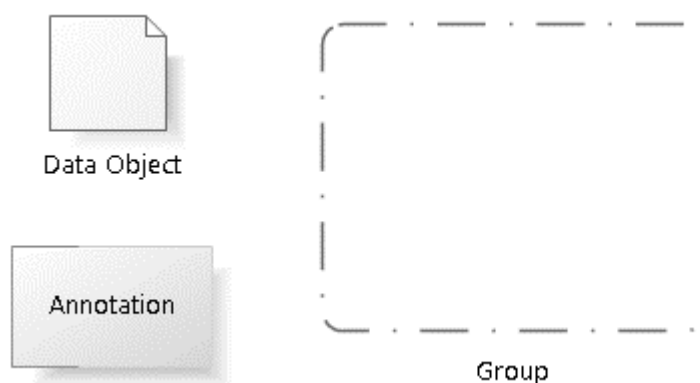


Figure 10 - Artifacts [author]

Data Objects gives information how documents, data and other objects are used and updated during a process. *Groups* are artifacts that group and highlight certain sections of a business diagram similarly like sub-processes with the main difference being that groups are not constrained by the behavior of Pools and Lanes. *Annotations* are used from modelers to provide additional information for the reader of a BPMN Diagram.

4.5. Designing a Business Diagram

In the previous segment we introduced basic concepts of BPMN and their graphical notations. A natural step forward is to use these concepts to design a business diagram in order to get a better understanding of business processing modelling and its use in real life. It is very important to start the design with a low complexity (simple) and then gradually add complexity levels when process gets more in depth.

For the purpose of this paper we will model a Credit Application process using business diagrams inside BPMN.

Process Description

This process starts with accepting the application of the client who is interested in acquiring a credit loan from a credit bank. Branch office then gives preliminary judgment if the client is suited for entering the loan application process. In the next stage, successful applicants are asked to deliver extra documents that need to be assessed by credit loan analysts. In order to not prolong the process indefinitely there should be a fixed period of time when applicant should submit the extra documents.

After the documents have been received, credit factory will analyze and conclude credit applications as accepted or rejected. Once a credit application has been accepted, back office will proceed with the disbursement of money and client will receive his loan and process will end.

Process Modelling

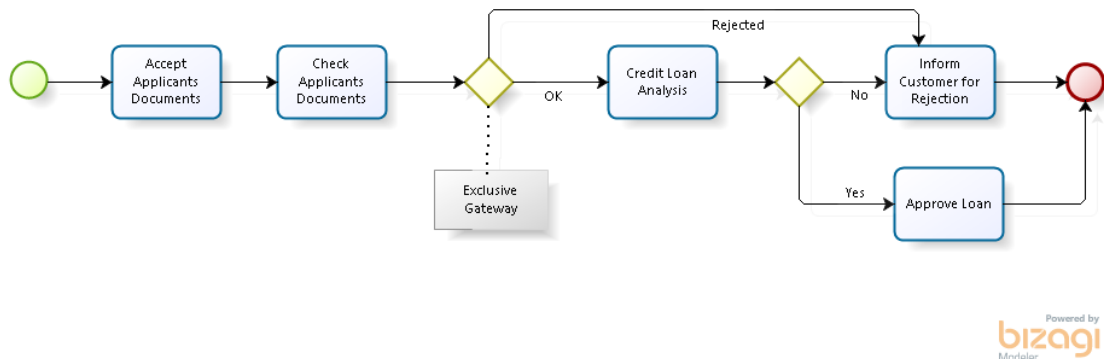


Figure 11 -Credit Application Process [author]

In this diagram we have designed the process description above. At the beginning the Start Event which is followed by two activities and then there is an Exclusive Gateway which means only one of the alternative paths can be taken. If applicant's documents are accepted then the sequence flow goes to the Credit Loan Analysis and then process goes through another gateway which acts similarly as the first one. However, if documents are rejected then the sequence flow goes straight to the end event and process ends there.

This process looks quite simple since its complexity level is very low but it is not very clear so there are activities that can be analysed in greater detail. Such is the second activity *Check Applicants Information*, which can be a process in its own.

Check Applicants Information sub-process is seen below.

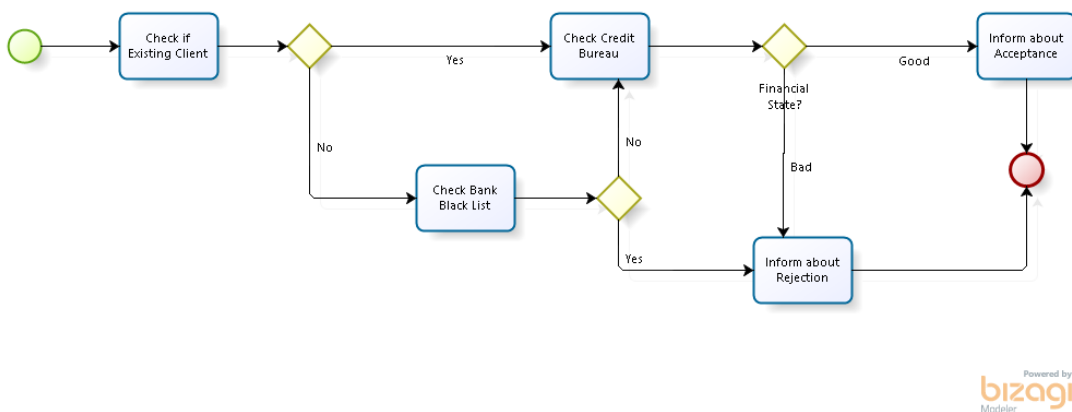


Figure 12 - Check Applicant Information Sub-Process [author]

Now the original process with the added sub-process looks like this

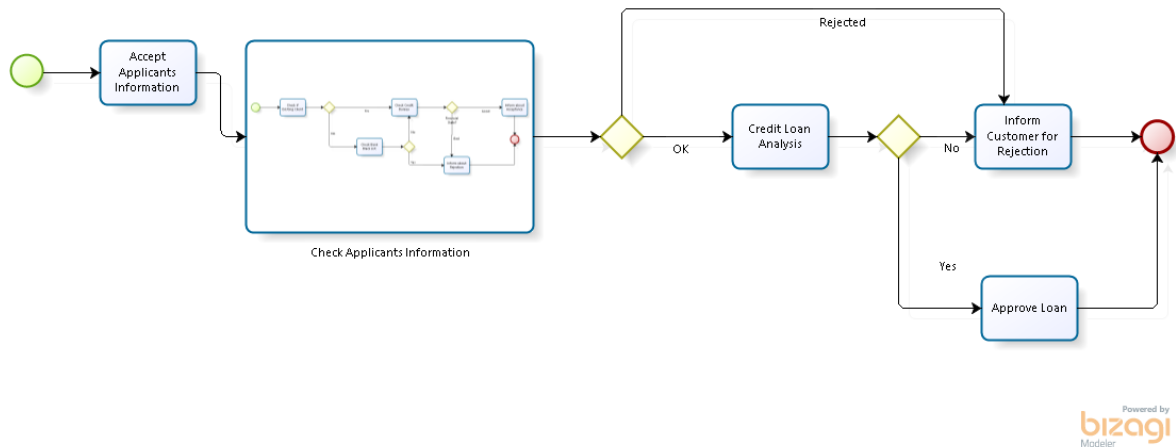


Figure 13 - Credit Application Process with added sub-process [author]

Once additional details are added to the diagram, process starts to become clearer and this gives modeller more confidence to change certain things which otherwise would have remained hidden.

Going back to the Credit Application process we see that at the moment the application is made, the applicant might not have included all the necessary documents and the current process doesn't really address this problem. Rejecting the whole application because of this issue is not very feasible since these kinds of issues are pretty common when applying for credit loans due to the high volume of documents required so there needs to be an activity that is more effective in handling this problem.

We can use a document receiving activity inside the current process but the fulfilment of this activity will depend on the applicant (external factor) and not on the bank employee (internal factor). This can be done by introducing an *Intermediate Event* for receiving documents. However, since this activity may or may not happen (applicant can bring, take too long to bring or not bring at all the extra documents), the process should account for all the outcomes and in BPMN this situation can be handled by *Event-Based Exclusive Gateways*.

The situation that needs to be modelled is as follows: The client has a fixed period of time to send the extra documents. If client doesn't send the extra documents in time, a contact is made with him to follow up the process. If he again doesn't initiate any action than the process terminates and the application is rejected. Nevertheless, if the documents are presented within the established time then the process will continue to the next activities.

The appropriate diagram for this process will be shown in the following model

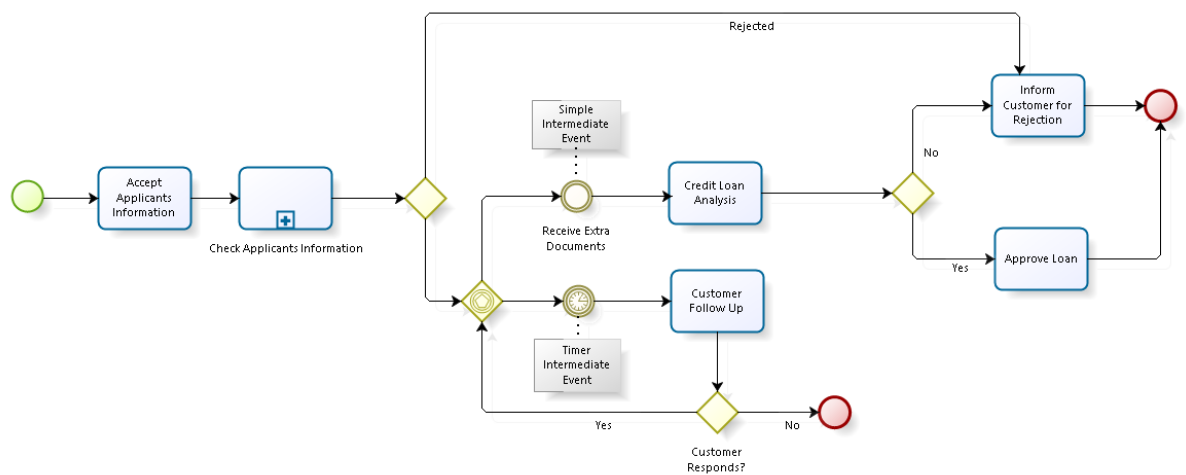


Figure 14 - Credit Application Process with event based gateways [author]

As we can see, the process looks more detailed now and even though there are many other activities that can be expanded, the purpose of this diagram is to show the idea behind the gradual improvement of the model which is essential when discussing quality metrics for business process modelling.

To finalize this diagram, we have to visualize the interaction between roles in this process. Roles are defined by their responsibilities in the process. In the current Credit Application Process we have three roles: *Branch Office*, *Credit Factory* and *Back Office*. All these roles will be sub-partitions of *Credit Bank*. BPMN uses swimlanes to visualize roles interaction in the process.

When involving roles into the process, the modeller should take into consideration the added complexity level that each role brings. Hence, in our Credit Application process, Credit Factory will be responsible for the whole process of Credit Loan Analysis and Back Office will be responsible for disbursing the money so both of these activities we will transform to sub-processes.

The adjusted diagram to fit the roles will be shown in the following model

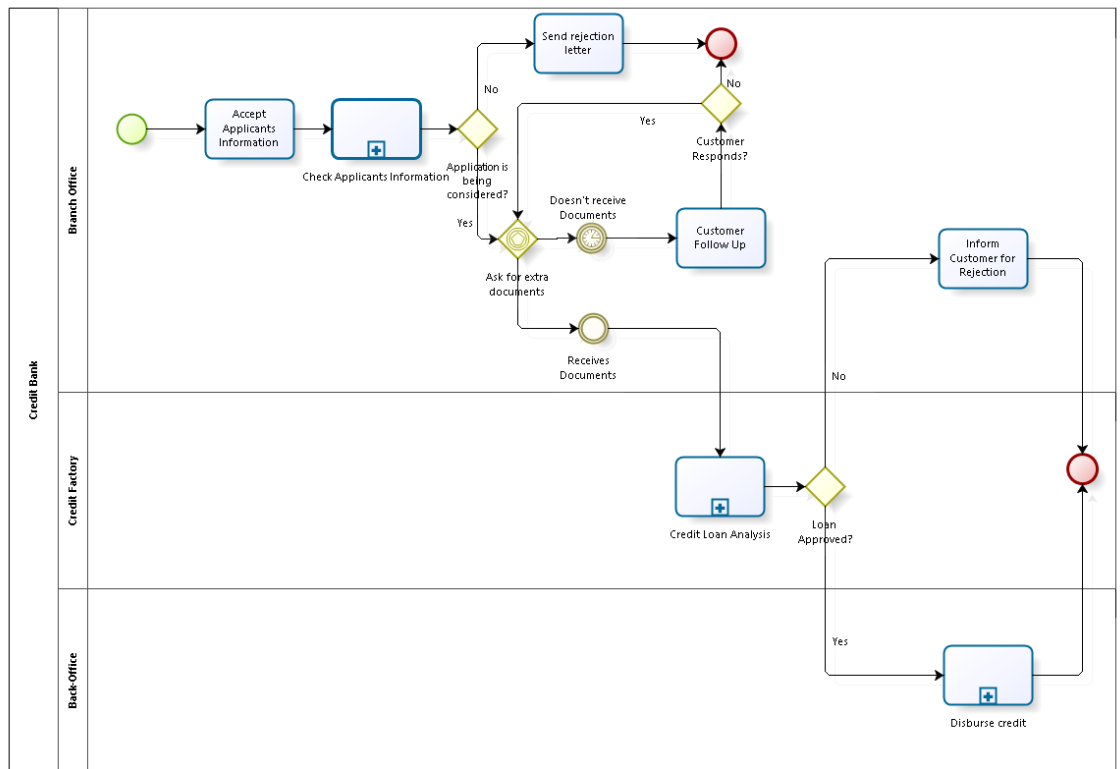


Figure 15 - Credit Application Process - final [author]

5. Quality Metrics for BPM – Current State

Being able to design good process models now has become the norm in order for businesses to stay competitive. A good business process model means that the model is reusable and can be improved moving forward. For bringing improvement to a process model which can lead to a total business process reengineering (BPR), there need to be some methods that are able to analyze and measure these models in order to apply the changes.

This has proven to be a challenge in this field since there is not a general accepted framework of model quality and measurement. So far quality has been associated with the understandability and maintainability of processes. For a process to have these qualities, it should not have high level of complexity in a way that business analysts do not find it impossible to track how one activity is affected by another. Once a process model is not easily tracable, it has a big chance of hitting unexpected errors thus failing to gain a substantial level of confidence.

This is a very similar process that we see also in software engineering. Program modules with high complexity tend to have a higher frequency of failures. A lot of research has been carried out to measure software complexity and a similar approach to give an empirical validation to business process models has proven to yield metrics that are able to measure the complexity of BPM (25).

According to a literature review in this topic done by members of Czech Technical University, up to 2015 they have estimated there are 22 measures of quality that are being used in Business Process Modelling (26).

The three most widely accepted metrics were shown to be:

- a) *Control-Flow Complexity (CFC)*
- b) *Number of Activities (NOA)*
- c) *Coefficient of network complexity (CNC)*

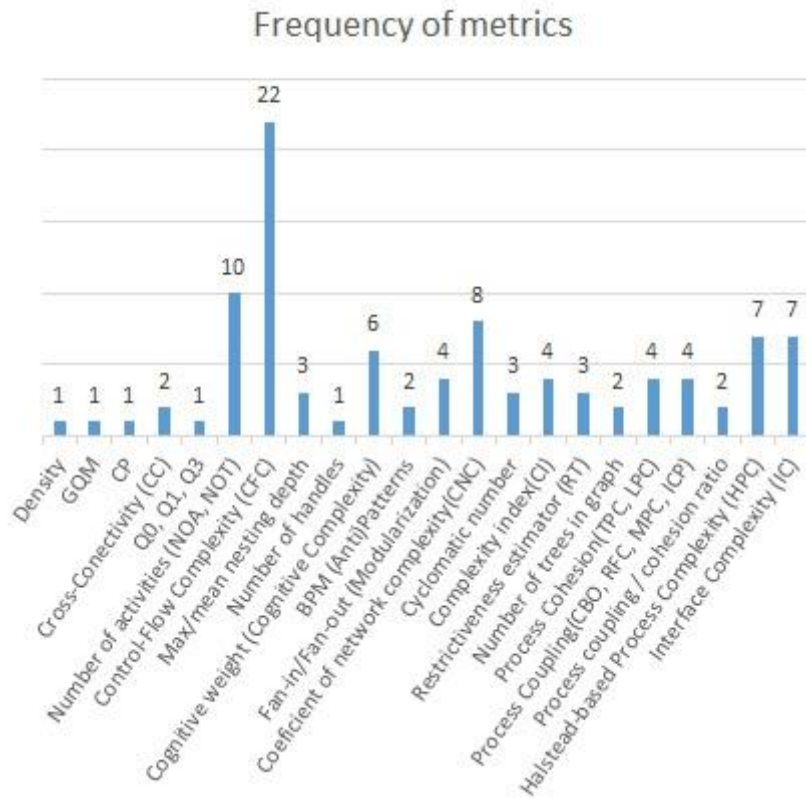


Figure 16 - Frequency of BPM metrics determined by CTU team (26)

5.1. Control-Flow Complexity (CFC)

Control-flow Complexity is a metric that was first empirically validated in BPM by J. Cardoso (25), who is the most notable source for process complexity measurements. The CFC metric is calculated in proportion with split structures that appear in a process. Split structures are very a good indicator of process complexity. The concept of CFC uses some techniques from McCabe’s cyclomatic complexity (27) in software engineering. The control-flow complexity formula for a process (P) can be calculated as follows:

$$CFC(P) = \sum_{i \in \{XOR\text{-splits}\}} CFC_{XOR\text{-split}}(i) + \sum_{j \in \{OR\text{-splits}\}} CFC_{OR\text{-split}}(j) + \sum_{k \in \{AND\text{-splits}\}} CFC_{AND\text{-split}}(k)$$

The overall complexity of the process is proportionate to the value of the CFC (P). The function of CFC (P) is dependent on XOR, OR, and AND splits.

In BPMN, splits are denoted by gateways which we have explained in the previous chapter.

- **XOR-split** corresponds to the *Exclusive Gateway* where only one of the outgoing paths can be taken when performing the process. In this case complexity is directly proportional to the number of paths that follow a XOR-split.

$$CFC_{XOR-split}(a) = fan-out(a)$$

Fan-out is a term, mostly used in digital electronics, that defines the maximum number of digital inputs which can be connected from the output of a logic gate.

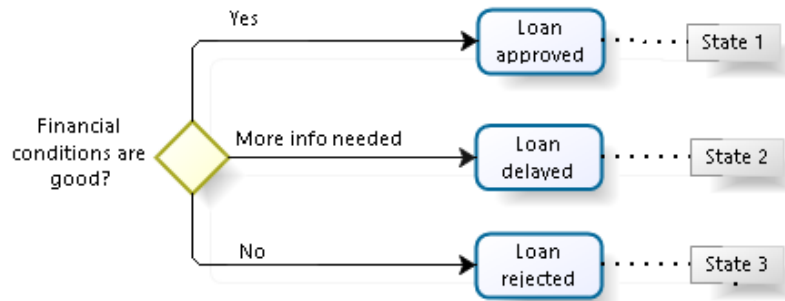


Figure 17 - XOR-split CFC [author using Bizagi Modeller]

- **OR-split** corresponds to *Inclusive Gateway* where n outgoing paths can be taken. The control-flow complexity for OR-splits is calculated as follows:

$$CFC_{OR-split}(a) = 2^{fan-out(a)} - 1$$

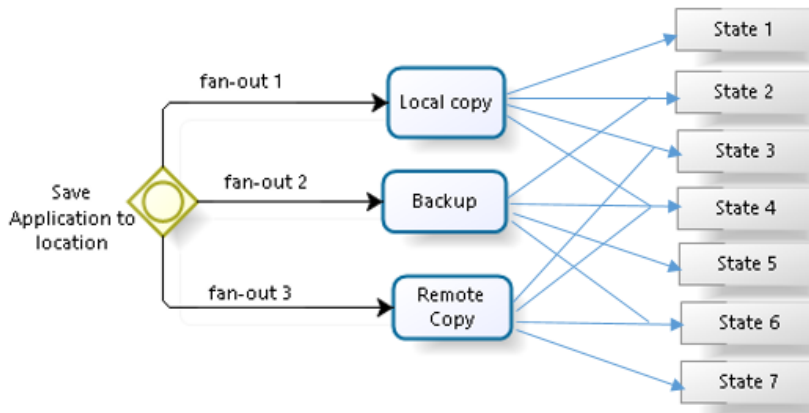


Figure 18 - OR-split CFC [author using Bizagi Modeller]

This means all the possible combination of outgoing paths with minimum one path followed should be taken into consideration. The condition where no path is followed isn't subject of an OR-split.

- **AND-split** corresponds to Parallel Gateway where all the outgoing paths are followed simultaneously and they all end up in one state.

$$CFC_{AND-split}(a) = 1$$

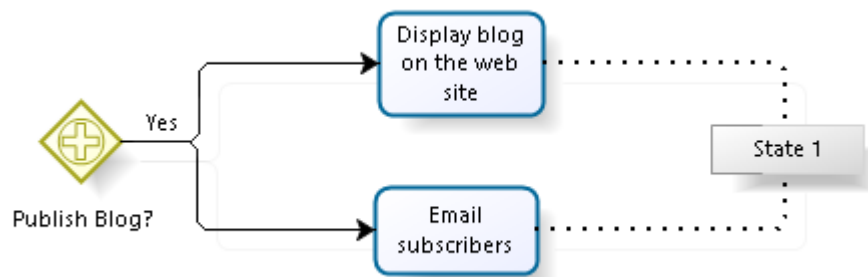


Figure 19 - AND-split CFC [author using Bizagi modeller]

Control-flow complexity (CFC) is very straightforward in understanding and implementing. The bigger the value of splits, the more complex is the process model. As seen above, each split formula is based on the number of states that proceed the construct.

A notable disadvantage of this metric is that it doesn't inform much about the structure of the model but rather counts the binary decisions that are possible to follow. Another concern for CFC is the fact that it tends to be skewed towards OR-splits since every additional activity in $CFC_{OR-split}$ will increase the sum exponentially thus making models that have a lot of OR-splits inherently more complex than the others.

5.2. Number of Activities (NOA)

One of the most basic and early measurements for software complexity is based on Lines of Code (LOC). The main concept behind this measure was that software program length can be a benchmark of program characteristics like error occurrences, reliability and maintainability.

This is considered to be an outdated concept in software development but in business process modelling a derivation of LOC has proven to be effective in measuring process complexity.

Cardoso et al (28) have derived three simple metrics from Lines of Code measurement. In business process these metrics are known as Number of Activities (NOA), Number of Activities and Control-flow elements (NOAC) and Number of activities, joins & splits (NOAJS).

NOA simply counts the number of activities in a process. This metric takes into an account merely the size of the process but not the functionality or complexity. Compared to the original LOC metric in software development, NOA is language independent which makes it way easier for users to understand it.

Since number of activities metric alone isn't very effective on differentiating between well-structured and not well-structured processes there was a need for another adaptation of LOC metric which besides number of activities it takes into consideration control-flow elements (NOAC). This adaptation then led to Control-flow complexity (CFC) which it is discussed in the previous segment of this paper. This proved to add more accuracy when measuring well-structured processes because control-flow elements influence execution sequences of activities.

For not well-structured processes that some modelling languages allow including Event-driven Process Chain (EPC) and Workflow nets, it is necessary to design a third metric because in these kind of processes splits do not necessary match a corresponding join. This metric is known as NOAJS. For this metric control-flow elements don't have any significance so they are replaced with joins and splits as separate evaluators.

5.3.Coefficient of Network Complexity (CNC)

Another field where complexity measures are proven to be effective when derived in business process modelling is graph theory. Since graphs are indispensable to modelling languages like BPMN, graph metrics have found implementation in measuring process models.

One of the most used graph metrics is Coefficient of Network Complexity (CNC). In 1966, Pascoe proposed the following definition for the Coefficient of Network Complexity (29):

$$CNC_p = \frac{A}{N}$$

A is the number of arcs and N is the number of nodes in the graph.

The concept behind CNC defines complexity in such a way that for a fixed number of nodes, a higher number of arcs results in an increasing complexity thus making the network (process) harder to grasp.

Deriving CNC in the context of business process model, the number of arcs has to be divided by the number of activities, joins and splits (NOAJS) (28).

$$CNC = \frac{A}{NOAJS}$$

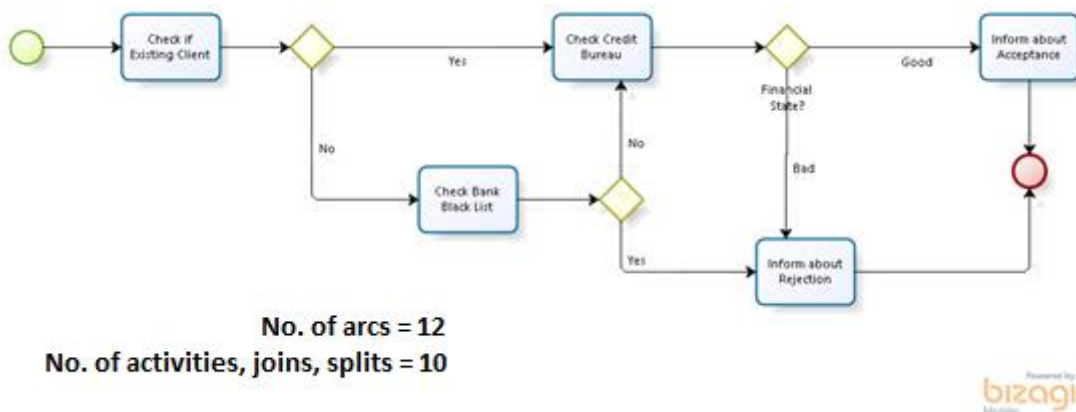


Figure 20 - Credit Application sub-process with added legend for CNC [author]

CFC, NOA and CNC can be the most used metrics when it comes to quality measurement of BPMs but since this is yet a not very researched field, there are a lot of other approaches and metrics that focus not just on the size and complexity of the process models. Modularity, coupling and cohesion are some other categories where researches are being done in order to develop new metrics and improve existing ones.

In the practical part of this paper we're going to introduce a new potential metric that can prove to be helpful in measuring complexity and quality in business process modelling.

Practical Part

6. Metrics Proposal - Feature Diagrams Compound Complexity

So far we have seen a number of complexity measures that are being used as metrics when evaluating various process models. Since there is yet no standardization of what is the best practice to value the quality of a model, the various researches have shown that there is still possibility to improve on the current metrics available.

We set our goal for this paper to try and find new potential measures that can be implemented as BPM metrics. Due to the interlinked nature of software programs and process models, we followed the principle of quantifying complexity into values that will enable designers minimize error rates and increase understandability of the model.

Following the literature review and metric classification done by Pavlicek et al. (26) where they have identified 22 measures of quality published between 1981 and 2014, we decided to see if there are new potential metrics published from 2014 up to January 2016 in related fields that can be added to this pool of existing metrics.

Of all the reviewed literature, we found one interesting publication in 2015 that measures complexity of Domain Models in Product Line Engineering (PLE) by using Feature Diagrams combined with ideas of Miller's, Metcalfe's and Keating's (30). Complexity measures proposed in this paper use cognitive complexity and tree-structure hierarchies with similar XOR, OR and AND connectors that are used in BPM, which make this measure very adaptable to structured business process models that are designed with BPMN.

By modifying certain aspects of cognitive and structural complexity which are more fitting to process modelling rather than domain models in PLE, we will show the consistency of this measurement compared to other metrics discussed earlier in previous chapters like Control-flow Complexity and Coefficient of Network Complexity.

6.1.Feature Diagrams in Domain Models

In the modern system design, design methodologies that adopt principles like information hiding, system decomposition and high abstraction level have been evolving for some time. A very good example of this is Product Line Engineering (PLE) or in terms of software products, this is known as Software Product Line (SPL).

A software product line (SPL) is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way (31).

This change of design from one single system to a shared group of system that inherit from one core system, paves the way for model-driven approach where the main focus is on high-level domain models.

Domain models, similarly like business process models, are used to describe certain information about the domain that are to be implemented in a software program. One of the notations used to present these properties are Feature Diagrams (FDs). One of the key reasons this model-driven approach prevails not just in SPL or PLE but also in the wider of context of software engineering is because it shows that dealing with complexity issues in stand-alone single programs is not enough anymore and the complexity problem should be tackled in a higher abstraction level.

Feature Diagrams are generally accepted models that represent domain instances in a high abstraction level. They are represented by connected graphs in which the root represents the domain, the intermediate nodes represent grouped alternative features, leaves represent feature variants and edges show the type of parent-child relationship between features. Feature model can be drawn up to n -level structure where n is the longest path from the root to the end-node. If the $i-1$ (where $root < i < n$) level node represents a variant point (a set of grouped features with shared properties) then i level nodes represent variants or terminals which are the child features of a variant point (30).

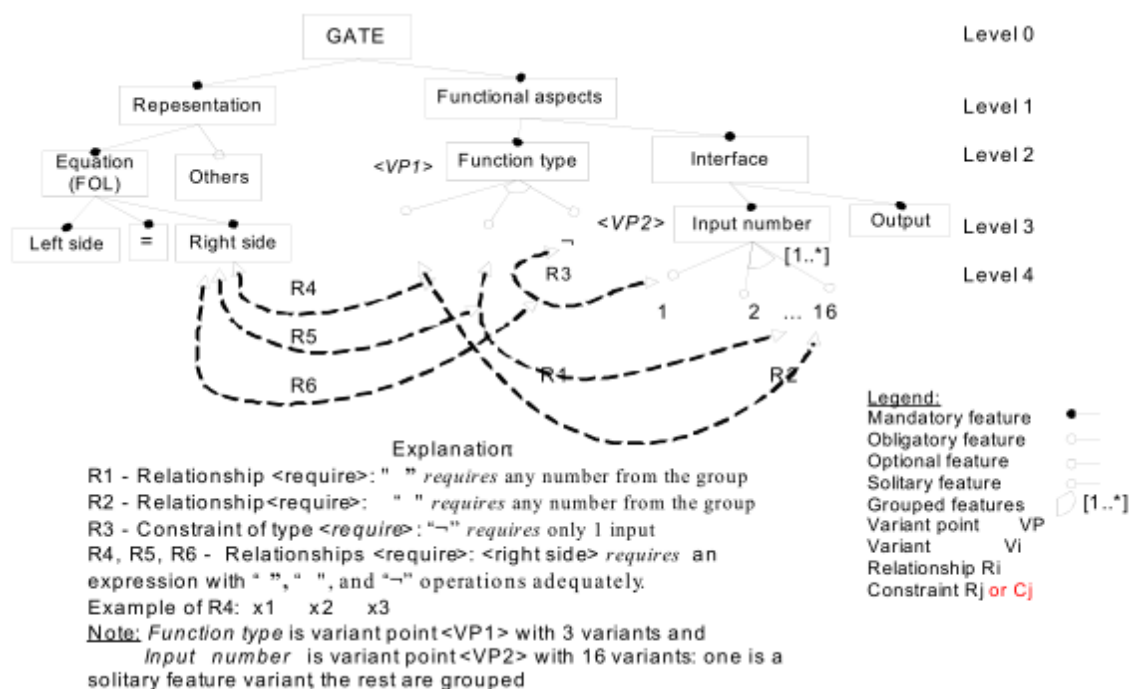


Figure 21 - A feature diagram of the GATE domain model [source: Stuikeys, et al. (30)]

It is easy to see that domain models can suffer from high complexity when there are added layers of features added to the model thus limiting the readability and clarity of the graph itself. This is in stark similarity with the complexity problems that appear in structured business process models when using BPMN.

6.2. Complexity Measures in Feature Diagrams

The main concepts behind designing complexity measure for structures like feature models or business process models deals with how much difficult is to understand the model. In this sense cognitive complexity plays a big role.

Stuikeys and Damasevicius (30) have used cognitive complexity combined with works of Miller and Keating (32) to give four empirical rules that can be applied to feature models.

Rules 1 and 2 deal with cognitive complexity, Rule 3 deals with structure complexity and Rule 4 deals with compound complexity. The aggregation of all these rules will tend to classify a feature model as low complex (simple), slightly complex, complex and overly complex (untestable).

Rule 1: Cognitive complexity is calculated as the amount of variant points in a Feature Diagram. The number of variant points must be 7 +/- 2 in order to prevent from high cognitive complexity if there are more than 9 variant points. The 9 variant points threshold, if exceeded, would be an indication that the user needs to divide the model into parts (in BPMN this would mean to include more sub-processes). If the number of variant points is less than 5 then value of the model is reduced due to the high amount of information hiding which leads to obsolete models.

Rule 2: Another measurement of cognitive complexity of a feature diagram is the maximal number of levels in the tree-structure hierarchy (vertical) or the maximal number of parts in each hierarchy level.

Rule 3: The structural complexity of a feature model is measured by the number of sub-trees where each variant point has only one selected variant. Every sub-tree or sub-model is derived from the initial feature diagram as a generic model for a certain domain. This rule has some correlation with cyclomatic number, another measure that defines the complexity of a program which has found good use in business process models as well.

Rule 4: This rule is a merge of cognitive complexity (Rule 1, 2) and structural complexity (Rule 3). This is known as a compound complexity, it is based on Metcalfe's empirical law which implies that the "power" of a network is equal to the square of the nodes while the "value" of the network is equal to the square of the edges of the network. Keating took Metcalfe's law and adapted it for the evaluation of complexity of a design partitioning (32). The Keating's measure is:

$$C = M^2 + I^2$$

C is the complexity measure, M is the number of models in a diagram and I is the number of interfaces between those modules.

Since this complexity measure can be presented as a graph using nodes and edges, it can be adapted as a metric in many other areas that use similar workflows like software programs, feature diagrams, business process models etc.

Stuikys and Damasevicius (30) have modified this rule for feature diagrams by taking into consideration cognitive weights of a feature to better measure the degree of the difficulty for AND, OR, XOR relationships when it comes to user's comprehension and understanding of the model diagram. This makes the consideration of this metric in BPMN even more relevant since the visual representation is key in measuring business process models.

By further following and adapting the complexity weights introduced by Shao and Wang (33), they have created the following table of weights for feature diagrams.




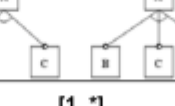

Feature Diagram element	Structure	Cognitive weight
Node (feature)		1
Mandatory feature relationship (and-relationship)		1
Optional feature relationship (or-relationship)		2
Alternative feature relationship (case-relationship)		3
Groupings of relationships (cardinality)	[1..*]	3
Relationships among nodes and constraint relationships (<requires>, <excludes>)		3

Figure 22 - Cognitive weights of Feature Diagram elements [source: Stuikys, et al. (30)]

Using this table of weights as a reference they adapted Keating's equation to fit the feature diagram model. The equation is estimated as:

Eq. 1

$$C_m = F^2 + \frac{(R_{and}^2 + 2R_{or}^2 + 3R_{case}^2 + 3R_{gr}^2 + 3R^2)}{\sum(\text{cognitive weights of relationship types})}$$

C_m - compound complexity measure,

F - number of features,

R_{and} - number of mandatory relationships,

R_{or} - number of optional relationships,

R_{case} - number of alternative relationships,

R_{gr} - number of relationship groupings (cardinality),

R - number of relationships among nodes

The denominator is the sum of cognitive weights which needs to equalize the role of relationships. This value for the denominator is based on Metcalfe's law criticism (34) which argues that the value of network in terms of complexity is not equal to the square of the number of connections but it is considerably less than this value.

One of the basic objections of Metcalfe's law is that not all connections (edges) will have the same weight like it is initially suggested but some nodes will communicate more with each other than the others, therefore connections (edges) value will vary.

The best illustration of the argument against Metcalfe's law about the value of the network being proportional to the number of users connected in the system (n^2), is the behaviour of network operators.

By Metcalfe's law, each network value is of the order of n^2 so if these networks decide to merge together they should have the value of the order $(2n)^2$ which is equal to $4n^2$, twice as more as the combined value of the networks $n^2 + n^2 = 2n^2$. From this simple arithmetical analysis, it is easy to see that the network operators would have every reason to merge since the value of their network would show a quadratic growth instead of a linear one.

Instead what happens is that network operators find it more feasible in competing with each other than joining forces, there are additional costs if users of one operator decide to call to the network of the other operator.

This shows us that Metcalfe’s law which was originally designed for telecommunications system doesn’t necessarily hold true.

We can extend this argument to other systems that use the concept of algorithms or generally nodes and edges similar like feature diagrams, UML or business process models. While there is an increase of value when new elements join the network or the model, the value is between linear and quadratic growth.

To address this issue to better quantify the use of the complexity metrics, the following inequality is presented which divides the square of model interfaces (edges) by the sum of the cognitive weights of those interfaces.

Eq. 2 – Calculated metric for value of Feature Diagrams

$$\left(\frac{(R_{and}^2 + 2R_{or}^2 + 3R_{case}^2 + 3R_{gr}^2 + 3R^2)}{\Sigma(\text{cognitive weights of relationship types})} < (R_{and} + R_{or} + R_{case} + R_{gr} + R)^2 \right)$$

$\Sigma(\text{cognitive weights of relationship types})$ – This sum accounts for the relationship types that are present in the model but it isn’t influenced by the number of relationship types. So for example, if there are 3 XOR elements, 2 AND elements, 5 OR elements, the sum will count only the weight of 1 XOR, 1 AND, 1 OR element.

6.3. Calculating Metrics for a Feature Diagram

We can use the GATE feature model shown in *Figure 21* as an example for which the complexity metrics satisfying the 4 rules above are applied. To have a clearer understanding on how relationship types interact in a feature diagram model below is a modified diagram of the one shown in *Figure 21*, including only elements that will be significant when discussing similar complexity metrics in BPMN.

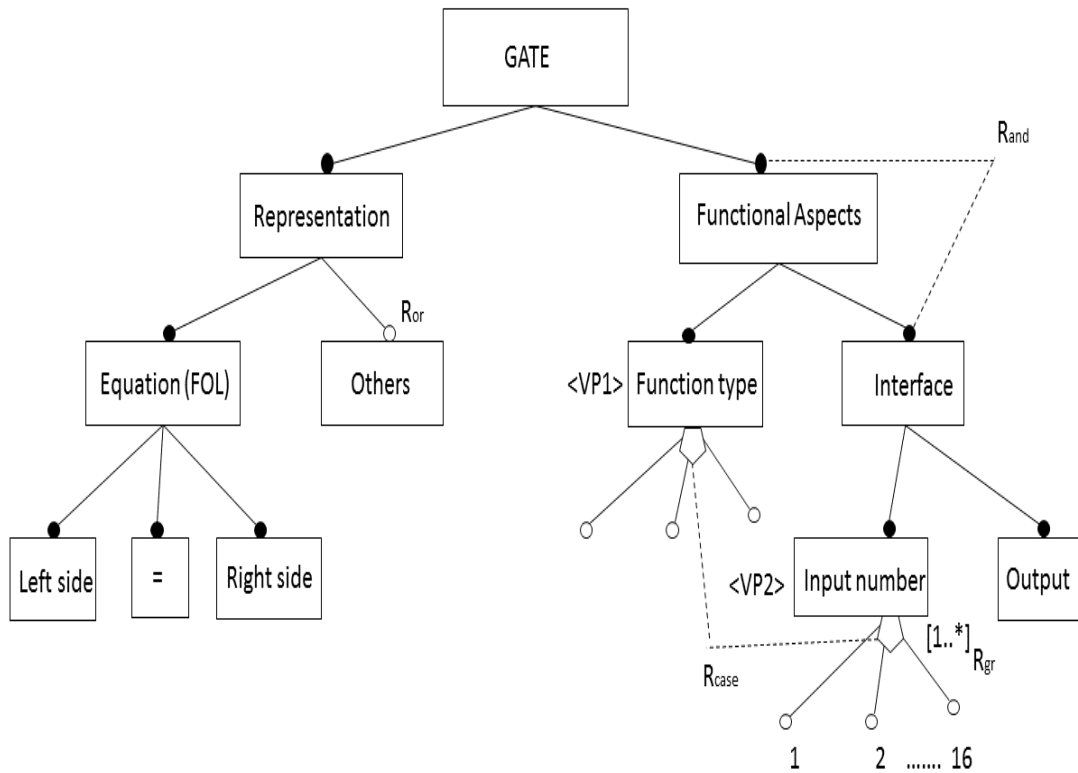


Figure 23 - GATE Feature Model with explained relationship types [author]

Edges that end with small black circles show and-relationships (R_{and}), edges ending with white circles show or-relationships (R_{or}) and edges that start with the pentagon in the diagram show alternative-relationships (R_{case}).

It is clear from the figure above that this feature model has maximum 5 levels in the tree-structure hierarchy, 2 features are classified as Variant Points <VP> which usually represent groups of features with shared properties. In total, there are 31 sub-models including the 16 nodes that are derived from the <VP2> *Input Number*.

Estimated complexity table for GATE model with corresponding weights for each rule is shown below:

Table 1- Estimated complexity table for CASE Feature Diagram [author]

Feature Diagram	GATE
Cognitive Complexity (Rule 1 – Variant Points)	2
Cognitive Complexity (Rule 2 – Hierarchy Level)	5
Structural Complexity (Rule 3)	31
Compound Complexity (Rule 4)	990
Variable Explanation	$F = 31; R_{and} = 10; R_{or} = 1; R_{case} = 4; R_{gr} = 1; R = 6$

To proceed on finding out C_m first we need to calculate the sum of cognitive weights of all the relationship types that are used in the model. Here we use the table presented in **Figure 22** to check which elements from that table are present in the GATE model. The following elements can be seen: AND-relationships, OR-relationships, CASE-relationships and Relationship Grouping or Cardinality element.

The sum of cognitive weight is shown in the table below:

Table 2 - Cognitive weight of FD elements presented in GATE model [author]

FD elements in GATE model	Cognitive weight
AND-relationship	1
OR-relationship	2
CASE-relationship	3
Relationship Grouping	3
SUM	9

The denominator value is proven to be 9 so now we have all the necessary variables we need to satisfy the Compound Complexity (C_m) equation.

$$\begin{aligned} C_m &= F^2 + \frac{(R_{and}^2 + 2R_{or}^2 + 3R_{case}^2 + 3R_{gr}^2 + 3R^2)}{\sum(\text{cognitive weights of relationship types})} \\ &= 31^2 + \frac{(10^2 + 2 * 1^2 + 3 * 4^2 + 3 * 1^2 + 3 * 6^2)}{9} = 990 \end{aligned}$$

From this calculation, we can conclude that we have found the value of *Compound Complexity*.

It is important to note that this particular metric which was derived from Rule 4 is of utmost importance when dealing with Business Process Models because it has all the necessary components to be a quality metric that will give a very good estimation of model's complexity. This will be discussed further on the next chapter.

7. Applying Feature Diagrams Compound Complexity in BPM

In the previous chapter we presented a potential new quality metric in the form of domain model complexity represented by Feature Diagrams. We found many similarities between domain models used in Product Line Engineering and Business Process Models. The research paper conducted by Stuijks and Damasevicius (30) opened a new door for considering this potential new metric in the BPM domain.

Potential interest can be shown in the *Compound Complexity*, a metric which is built in the work done by Miller, Metcalfe and Keating in designing complexity measures throughout various fields spanning from telecommunication networks to design partitioning. Since their essence of their work uses graph theory, algorithms and other computational concepts, there is empirical ground to attempt to prove that this approach can help measure business process models in a new way.

In this chapter, the author will try to adapt the domain model complexity presented in the last chapter in the world of business process models. Although, there are many complexity metrics already in use in BPM, the introduction of *Compound Complexity* as a new metric aims to prove that this metric can offer a balanced quantification between the number of activities and other flow objects that interact with them.

There is a necessity to modify the cognitive weights presented to be more fitting for business process models and this will be followed by different tests concluded between different BPMs in order to prove that the metric is stable and is sensitive enough to detect fluctuations in model difficulty.

For ease of use, visualization and better understanding of processes, BPMN notation will be used. The software used to design business diagrams is Bizagi Modeler.

7.1. Setting Cognitive Weights for BPMN Elements

When discussing the comprehension of a given model, there is a natural inclination to focus on the architecture of that model as a cornerstone of measuring difficulty or complexity. In visualized systems where we can draw a diagram such is BPMN notation this inclination is even more obvious.

In order to grasp and quantify this visual complexity, we use cognitive weights to measure certain elements that seem more complex than the others or vice-versa. Cognitive weight is the degree of difficulty or the amount of time and effort that is needed to comprehend a given model (33).

In software development, there is a set of control mechanisms known as basic control structures (BCSs) of software (35). They are used to build and increase complexity of a software architecture. Three most used BCSs are: sequential, branch and iteration structures (36). The cognitive weights were first used to measure these structures

Similarly, we could see in the previous chapter when discussing the complexity of domain models that this concept of BCSs is prevalent albeit modified to fit the feature diagrams structure. One stark similarity that we mentioned between domain models and business process models is that they have similar type of “control structures”. In Feature Diagrams notation these are known as connectors while in BPMN notation they are called Flow Objects.

From Fig 6.2 we can already see what cognitive weights are set up to measure Feature Diagrams (FDs) complexity and since the connectors (and, or, case) are mirrored by BPM flow objects (AND, OR, XOR) it looks very logical to adapt these exact cognitive weights as part of our metric evaluation in BPM.

However, it is very important to make a distinction between the role of domain models and the role of business process models since this can influence cognitive weights significantly.

A significant logical difference between these two disciplines is that domain models deal mostly with systems that are inherited from previous systems while adding new features along the way. Business Process Modelling (BPM) primarily concerns itself with description and modelling of an end-to-end process. While multiple processes that are dependent on one another can be constructed and reusability is highly encouraged, this is an advantage rather than the norm.

There are other factors that contribute to this division, such as the importance of branching in BPM and the crucial role that it plays in comprehending complexity in a model. Branching itself does not play such a big role in Feature Diagrams. An AND-relationship that has two branches, has the lowest cognitive weight (cog weight = 1), equal to the weight of a simple sequence that connects two nodes.

Taking all this into consideration, we consider that it is necessary to construct a new table of cognitive weights, one that would be more suitable to address business process models and more concretely to appeal the BPMN notation since that is the notation that is most widely used when it comes to designing process models.

There have been previous attempts to define cognitive weights for BPM (37), however these attempts were mainly concerned with creating a single metric that uses only cognitive weights as a unit of complexity measurement. While certainly this single metric has found a wide use, it extends more to the general concept of BPM by not favouring any notation standards.

In our newly proposed metric – Compound Complexity, we are trying to combine different metrics like Number of Activities (NOA), Cognitive Weights (CW) and using the ideas of Metcalfe & Keating for measuring a network, to come up with a suitable metric that can be easily calculated when using BPMN notation.

For this reason and for the purpose of this paper, we introduce a new table of cognitive weights suited more for BPMN notation where besides gateways and activities, it takes into consideration other flow elements like Start, Intermediate and End events.

The BPMN adjusted table of cognitive weights looks like below:








Flow Object	BPMN	Branching	Cognitive Weight
Activity		No	1
Start Event		No	2
End Event		No	2
Intermediate Event		No	2
Exclusive Gateway (XOR)		$n = 2$	2
		$n > 2$	3
Parallel Gateway (AND)		$n = 2$	2
		$n > 2$	3
Inclusive Gateway (OR)		$n = 2$	4
		$n > 2$	5

Figure 24 - Cognitive weights for BPMN elements [author]

The simplest interaction in a model is the *Activity* sequence flow, it has one source and only one target, so unanimously this has the Cognitive Weight = 1 in all similar systems.

Following Activity, the next lowest weight goes for Event flows which have Cognitive Weight = 2. Although their notation is self-explanatory, every event has some specific role which accounts for extra added weight. *Start* indicates where a process will start and thus won't have any sequence flow connecting to it. *End* indicates where a process will terminate and thus won't have any sequence flows going out of it. *Intermediate* indicates where something happens between Start and End so while it won't start or terminate a process, it does affect it.

Finally, there are the cognitive weights for Gateway flows or else known as Decision flows. There are many scenarios that need to be assessed when measuring cognitive weights for decision flows. As we mentioned earlier, branching plays a certain role in BPMN so it is important to pay attention the branching span of the flow.

Independent of the decision flow used, a Gateway that has only two outgoing paths ($n = 2$) is not so difficult to comprehend. In fact, that represents the minimal number of outgoing flows so it is logical to use a low cognitive weight. For *Exclusive (XOR)* and *Parallel (AND)* Gateways, we can apply this logic since in a two-branch decision flow both these gateways will execute only once. *XOR* flow will choose one path and execute it while *AND* flow will execute in parallel both paths.

We can argue that even when a Gateway has more than two outgoing paths ($n > 2$), again both these gateways will execute only once. *XOR* flow will choose one of the n paths while *AND* flow will execute in one take all the n paths in parallel. However, the factor of visual comprehension should not be neglected. A model that has many branches will be more intimidating for the user that the one with less branches, so an increased cognitive weight is reasonable. In this case, when $n > 2$, cognitive weight for *Exclusive (XOR)* and *Parallel (AND)* is 3.

A special case of Gateway is the *Inclusive (OR)* Gateway which has a more complex functionality so the previous weights discussed for *Exclusive (XOR)* and *Parallel (AND)* are not really valid here. This is because regardless the number of outgoing paths (n), an OR flow can execute more than once. As discussed in Chapter 5.1, the number of states an OR flow can execute is $2^n - 1$.

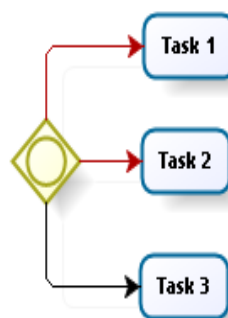


Figure 25 - An OR flow with n=3 [author]

Although it is very obvious that the complexity of an OR flow is not increasing in a linear way towards an XOR flow but it is rather an exponential increase, still we have to take into consideration the redundancy this element creates in the visualization of the model in a BPMN diagram.

A model that might be designed using 4 XOR gateways, could be designed using only 1 OR gateway thus decreasing the complexity and this is the most important thing when discussing the quality of business process modelling. For this reason, when there are two outgoing paths ($n=2$), the estimated cognitive weight for XOR is 4 (it mirrors a double increase in the XOR flow weight).

If the number of outgoing paths is bigger than two ($n > 2$), the estimated cognitive weight is 5. This increase of weight by only 1 unit is simply because in case of large process models, it is encouraged to use the OR flow because of the element redundancy it provides in a BPMN diagram.

7.2. Metric Calculation

Once we have defined the cognitive weights that are more relevant to the BPMN use of elements, we can move forward to make arguments for a suitable equation that will describe compound complexity.

The initial foundation for this metric is based on the combination of work done by Keating's assessment of Metcalfe's law in design partitioning (32) and the recently published paper by Stuijks and Damasevicius (30) that proposes this approach to measure complexity in domain models by using feature diagrams.

The Rule 2 of Keating's design partitioning complexity says (32): "To calculate the complexity of a design partitioning, add the square of *number of modules* (or *nodes* on the graph of the design) to the square of the *number of interfaces* between modules (or *edges* on the graph of the design). "

From this rule, it is quite clear that we can use this calculation to measure business process models by using a simple analogy:

Number of Modules ↔ *Number of Activities*

Number of Interfaces ↔ *Number of other flow objects*

“Other flow objects” in this case it means all the flow objects that are not activities (or sub-activities).

An illustration of the analogy between design partitioning and business process modelling is shown in the **Figure 26**.

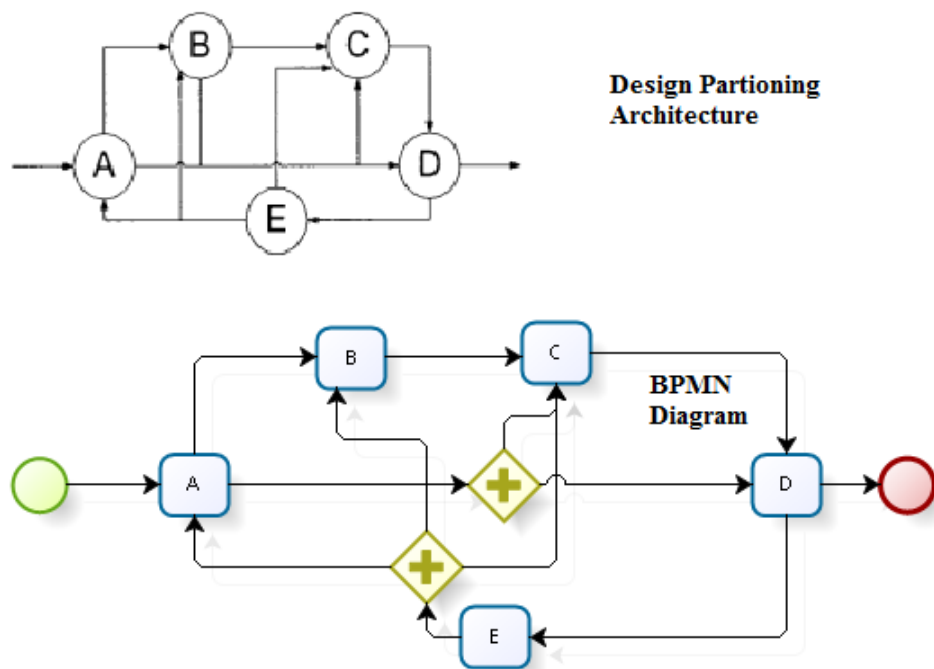


Figure 26 - A design partitioning model mirrored by BPMN [author]

Once we develop the analogy, than we can continue to construct an initial metric

Eq. 3 - Feature Diagram Compound Complexity

$$C_m = (NOA)^2 + (\text{Number of other flow objects})^2$$

C_m – Compound Complexity

NOA - Number of Activities (already existing metric described in Chapter 5.3)

This equation is valid if we take into account the original Metcalfe's Law that the power of network is equal to the square of nodes in it while the value of this network is equal to the square of the amount of edges in it. If we translate this into the BPMN language, it refers to the equation (Eq.3).

However, as we discussed in the previous chapter, this law isn't really valid anymore since the value of the network is considered to be less than the square of edges (34) so we need to find a more appropriate estimate for evaluating the value of a network, or in our case, a process model.

This issue has been highlighted also in the paper discussing compound complexity in domain models (30) and it has been adjusted to reflect a value that is less than the square of nodes. What has been proposed there indeed is the equation (Eq. 2).

After we have created our own table of cognitive weights for BPMN elements, we can easily adjust the second part of the equation (Eq. 3) in a formula suited for calculating compound complexity for business process models.

Eq. 4 - Proposed equation for the "value" of BPM

$$\frac{(C_w R_{start}^2 + C_w R_{end}^2 + C_w R_{intermed}^2 + C_w R_{xor}^2 + C_w R_{and}^2 + C_w R_{or}^2)}{\Sigma(\text{cognitive weights of relationship types})}$$

C_w – Cognitive Weight

In this equation, cognitive weight of each BPMN flow object multiplies the square of the total number of the flow object that is presented in the model. It is important to keep in mind that the number of outgoing paths (n) influences the cognitive weight for gateways (decision flows).

We tested this formula to see if it is producing good results on business process models built in BPMN but we saw that for initially low complex models the results are very biased towards the first part of equation which represents the number of activities.

We used the models we designed in Chapter 4.5 when we were describing a Credit Application process model to test this potential metric. The first model is a simple one with only 5 activities, 2 XOR gateways ($n=2$), 1 start event and 1 end event.

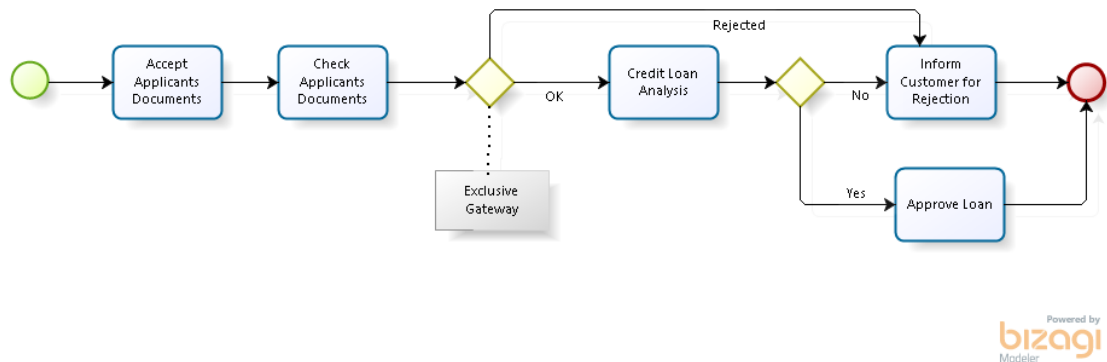


Figure 27 - A simple business process model [author]

$$C_m = NOA^2 + \frac{(C_wR_{start}^2 + C_wR_{end}^2 + C_wR_{intermed}^2 + C_wR_{xor}^2 + C_wR_{and}^2 + C_wR_{or}^2)}{\Sigma(\text{cognitive weights of relationship types})}$$

$$C_m = 5^2 + \frac{(2*1^2+2*1^2+0+2*2^2+0+0)}{(2+2+2)} = 25 + 2 = 27$$

From here it is clear that the second part of the equation which has value 2 has very little influence compared to the first part which has value 25. It is obvious that for simple models like this, this formula is skewed heavily towards the *NOA*.

Let's see how this formula behaves in a slightly more complex model that uses an expanded sub-process. Although sub-processes are a good way on reducing the visual complexity of a model, for the sake of this example we have expanded it just for measurement purposes.

This business process model (including elements inside the sub-process) has 10 ACTIVITIES, 2 START events, 2 END Events, 2 INTERMEDIATE Events and 4 XOR gateways (n=2).

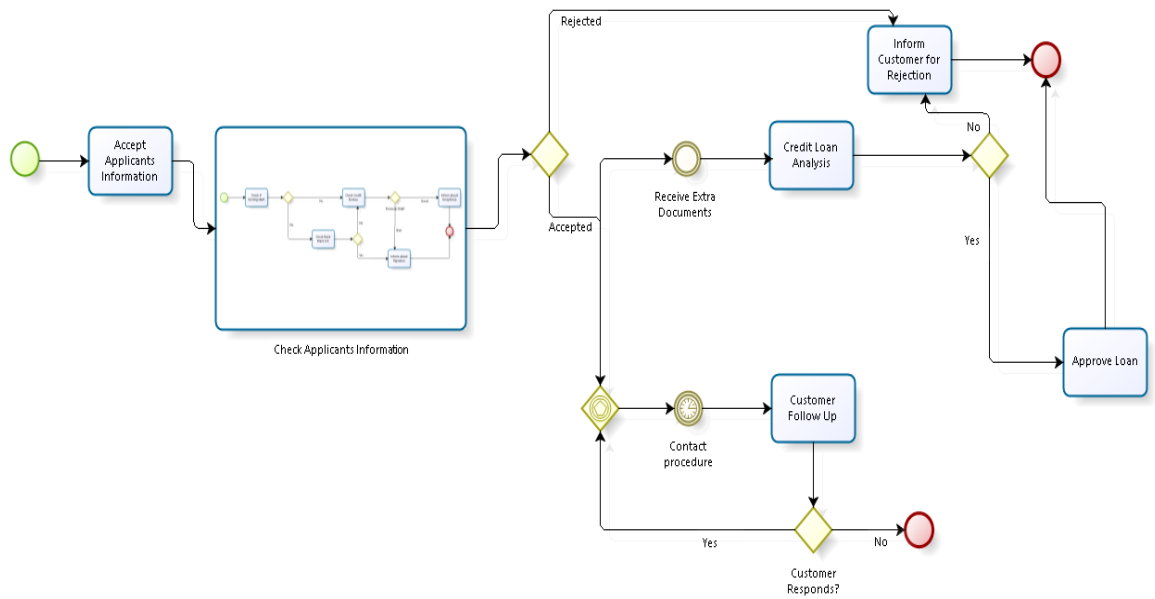


Figure 28 - A more complex business process model [author]

$$C_m = 10^2 + \frac{(2*2^2+2*2^2+2*2^2+2*4^2+0+0)}{(2+2+2+4)} = 100 + 5.6 = 105.6$$

Again we can see that similarly like it was the case with the simpler model, the result is heavily skewed towards *NOA* with only a fraction of the result accounting for the other flow objects like gateways and events.

From these two examples we can conclude that this metric is not really balanced and at this point is proving to be a very similar metric like *NOA* and although this metric can be more feasible in measuring complexity for domain models like we showed in the previous chapter, it is not the case with business process modelling.

If we get back to the original concept that refutes Metcalfe’s Law (30)(35), it is implied that the “value” of the network is less than the square of edges however from these two examples and many other tests, the second part of the equation which represents the “value” of the model in some cases it is showing to be even less then the linear growth.

While the value of a network is accepted to be lower than the quadratic value of Metcalfe's Law, there are several quantitative arguments implying that the value of the network is larger than the linear growth of order p (38).

A good observation that fits between the quadratic and linear growth in a network is made by A. Odlyzko and B. Tilly who propose that the network growth is proportionate to the function $p \log(p)$ where p is the size of the network. This explains why connectivity has more value than content (thus larger value than linear growth) but in the other hand this is only slightly faster than the linear growth which explains why large interconnection requires considerable time and effort to achieve (38) (39).

As long as complexity in business process modelling is concerned, this function looks to be a better fit than the equation (Eq. 4). Since the nature of process models is slightly different from computer networks and usually models are inherently smaller in size than networks, it makes sense to use to adjust the function $p \log(p)$ to the natural logarithm $p \ln(p)$.

A more important reason that we switch to natural logarithm (\ln) is that for the less complex models we can't be sure \log function will yield higher values than linear complexity. Among other things \ln is unambiguous compared to the \log which is used with both bases 10 and 2. It is worth noting that \ln yields higher values which can balance out the *NOA* bias we encountered in the previous examples.

When we talk about p , we are talking about the cognitive weight of other flow objects (not including Activities) which now is removed from the squared power

Eq. 5 - Flow Objects Cognitive Weight

$$p = (C_w R_{start} + C_w R_{end} + C_w R_{intermed} + C_w R_{xor} + C_w R_{and} + C_w R_{or})$$

The adjusted $p \ln(p)$ function for our complexity measure is noted as:

Eq. 6 - Proposed equation for the "Value" of the BPM

$$p \ln(p) = (C_w R_{start} + \dots + C_w R_{or}) \ln(C_w R_{start} + \dots + C_w R_{or})$$

To measure how $p \ln(p)$ (Eq. 6) holds when compared to the linear complexity, FD complexity (Eq.2) used in feature diagrams and $p \log(p)$ we have compiled a chart how these functions behave towards p which has minimum outgoing paths ($n=2$).

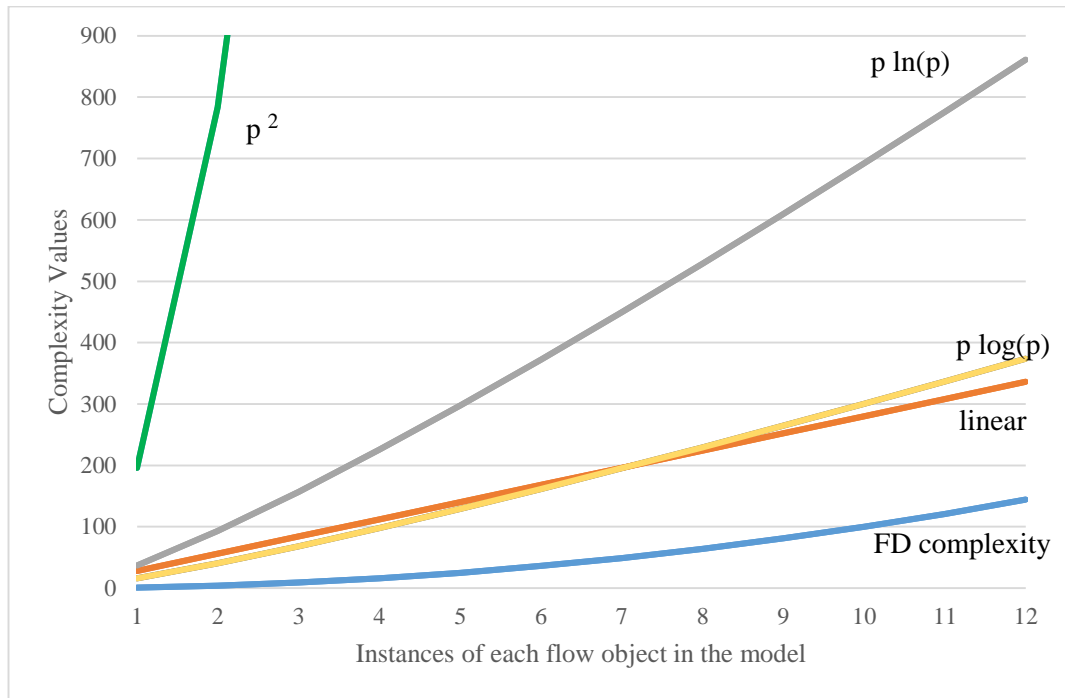


Figure 29 - Complexity growth between functions [author]

In *Figure 29*, we have imagined a scenario where all object flows appear in the same quantity in the model so it would be easier to see how complexity behaves when we increase the number of flow objects (x-axis).

Our desired area for measuring the value of the model as mentioned earlier lies between the linear $2p$ and quadratic p^2 complexity.

What this chart shows us is that *Feature Diagram (FD) Complexity* we considered earlier is retrieving results that are way under our desired area curve so we can dismiss this metric from our further evaluation.

Regarding our other consideration for measuring value of the model, $p \log(p)$ function is proving to surpass the linear complexity value only after a certain increase in

model complexity. This curve behaviour confirms our earlier observation that $p \log(p)$ function does good in larger complex models (networks) but not so well in smaller, simpler ones.

This leaves us with the $p \ln(p)$ function which from the beginning of the curve is showing results in the desired area and as model complexity increases, complexity values are closer to linear growth than the quadratic one, so it fits the estimate presented in Odlyzko and Tilly research paper (38) that complexity value of a network is slightly larger than the linear growth. We can see that this estimate retrieves pretty good results when evaluating the complexity of business process models as well.

Now, we can go ahead and improve our Compound Complexity metric with the following equation:

Eq. 7 Full equation for Compound Complexity (BPM)

$$CC_{BPM} = NOA^2 + (C_w R_{start} + \dots + C_w R_{or}) \ln(C_w R_{start} + \dots + C_w R_{or})$$

CC_{BPM} – The new notation for Compound Complexity in order to differentiate from FD Compound Complexity.

We can test now the previous model presented in **Figure 27** (5 ACTIVITIES, 2 XOR gateways with n=2, 1 START and 1 END event)

$$CC_{BPM} = 5^2 + (2 + 2 + 2 * 2) \ln(2 + 2 + 2 * 2) = 25 + 16.63$$

$$CC_{BPM} = 41.63$$

It is clear to see that compared to the previous FD complexity metric, the second part of the equation now has more significant value and it diminishes the bias towards the *Number of Activities* complexity value.

Let's see how this redesigned Compound Complexity (BPM) behaves on the more complex model shown in **Figure 28** (10 ACTIVITIES, 4 XOR gateways with n=2, 2 START, 2 INTERMEDIATE and 2 END events).

$$CC_{BPM} = 10^2 + (4 + 4 + 4 + 8) \ln(4 + 4 + 4 + 8) = 100 + 59.9$$

$$CC_{BPM} = 159.9$$

Even though the model complexity has increased, the second part of the equation doesn't show to be skewed and it maintains the balance between the "power" (*NOA*) and the "value" of the model ($p \ln p$).

So far, we have shown empirical validations of our newly proposed metric but we have yet to be sure this metric can quantify the complexity in terms of comprehension and understanding. This is very important in order to firmly establish this new metric as a credible one for measuring quality in business process modelling

In the next chapter, we will show how this metric is able to differentiate between two similar models and how it holds when compared to other existing metrics.

8. Metric Testing and Comparison – Case Study

In the previous chapter we tried to apply the FD Compound Complexity metric used in domain models. We found that although this metric has good premises we had to do some adjustments. First, we had to redefine cognitive weights so they would be more suitable for BPMN language and second, we replaced the second part of the equation with $p \ln(p)$ function which responds better to the requirement of model's value being between linear and quadratic complexity.

Although we were able to see a more tuned metric after applying these changes, we need to reach a higher confidence level about the measurement of complexity, especially when faced with real life business process models. It is important that this metric has the ability to challenge the designer to create better, more understandable process models when applicable.

To achieve this, we will design two business process models for a real life application called FactOrEasy, a management game developed by the supervisor of this thesis Mr. Josef Pavlicek. The concept of this case study is to measure the BPMs as they get designed, from the simple functionalities to the full process model, and try to prove that our *Compound Complexity (BPM)* metric is able to differentiate between the slightly less complex model and the other slightly more complex model.

To further strengthen the case for our metric, we will see how other existing complexity metrics are evaluating these models and how they compare towards our metric. This will give us a clearer picture how well *Compound Complexity (BPM)* can establish itself as a good quality metric in the array of other quality metrics out there.

8.1.FactOrEasy Game Overview

FactOrEasy is an online management game (40), targeted mostly to management students but it can be played by anyone who is interested in topics of investment, retail, and logistics.

This application was backed up by TAČR (Technologickou agenturou České republiky) as part of the program to support applied social science research and experimental development OMEGA 2013⁴.

Concept of the game

There are 4 players in the game; 1 human user and 3 robots. Each player starts with a fixed budget. There is a Material Market where players can buy materials available to create products. Material's minimum price are randomly set by the market. To create a product from a material, player needs to have a factory. Only 1 product can be created from 1 factory. There is also a Product Market where players can sell their products to fill the market demand. Similarly, the maximum price a product can be sold is also set by the Product Market.

FactOrEasy version 2.0


GAME STATUS	
Game Round	1 ▲
Number of Players	4 ▼

COSTS WINDOW			
Material Storage Costs (per unit)	300	Fixed Fact. Costs	1000
Product Storage Costs (per unit)	500	Production Costs	2000
Sum of Purchased Material(s)	0	Periodic Payment	0

MATERIAL MARKET	
Material Available	5 ▲
Minimum Possible Price	458 ▼

PRODUCT MARKET	
Product(s) Demanded	5 ▲
Maximum Possible Price	4918 ▼

COMPETITORS WINDOW				
	Human	Robot 1	Robot 2	Robot 3
Cash	5800	5800	5800	5800
Material(s) in Stock	4	4	4	4
Product(s) in Stock	2	2	2	2
Material(s) Demanded	0	0	0	0
Offered Price for Material (per unit)	0	0	0	0
Purchased Material(s) (units)	0	0	0	0
Product(s) Offered (units)	0	0	0	0
Sold Product(s) (units)	0	0	0	0
Price per one Product	0	0	0	0
Sales	0	0	0	0
Loan	0	0	0	0
Number of Factories	2	2	2	2



DECISION MAKING WINDOW	
Material(s) Demanded	<input type="text"/>
Offered Price for Material (per unit)	<input type="text"/>
	<input type="button" value="Skip"/> <input type="button" value="BUY MATERIAL"/>
Product's Units Requested	<input type="text"/>
	<input type="button" value="Skip"/> <input type="button" value="PRODUCE"/>
Product's Units for Sale	<input type="text"/>
Selling Price (per unit)	<input type="text"/>
	<input type="button" value="Skip"/> <input type="button" value="SELL"/>
	<input type="button" value="Next Round"/>
	<input type="button" value="Factory Request"/> <input type="button" value="Loan Request"/>

Figure 30- A screenshot of FactOrEasy game [source: TACR Program Omega]

⁴ The whole License Agreement (in Czech) can be accessed here <http://factoreasy.cz/FactOrEasy/APP/connector/0/0/licence/Smlouva.pdf>

The game is won from the player who accumulates the highest budget in a span of 12 Rounds. Each player initially starts with 2 factories, 4 Materials and 2 Products in their stock. There are maintaining costs for holding stock and also there is factory cost which gets calculated after the end of each round.

If after the end of each round player's budget falls under 0, the player goes bankrupt and the game is lost. Also a player loses if after the end of 12 rounds, he doesn't have the highest capital.

There are fixed loans which a player can take anytime during the game process but after each round the player has to pay a periodic payment to clear out his loan. A player can purchase factories which will enable him to create more products from materials but also increase his factory costs.

These were some of the basic rules which will help to better understand the design process. A more detailed description of rules and flow of the game can be found in the user manual⁵.

8.2.FactOrEasy Model Design

There are 3 basic flows how the player can interact with the system within a round. Player can choose to:

- Buy Material or Skip
- Produce or Skip
- Sell or Skip

A player can choose either of these 3 paths and he can use more than 1 path within a round. Alternatively, he can choose neither of them and basically skip all the flows and go to the next round. Part of the constraints is that the flows should be executed in the same order as described above.

⁵ Link http://factoreasy.cz/FactOrEasy/APP/connector/0/0/help/FactOrEasy_manual_TACR.pdf

There are two ways how we can start designing this model:

1. *FactOrEasy Model 1 (partial)* – we use 3 XOR gateways with minimum outgoing paths (n=2) to go through each flow.
2. *FactOrEasy Model 2 (partial)* - we use 1 XOR gateway (n=2) and 1 OR gateway with three outgoing paths (n = 3). Three OR paths will represent each flow and the XOR path enables the player to skips all three flows.

The designed *FactOrEasy Model 1 (partial)* looks like below:

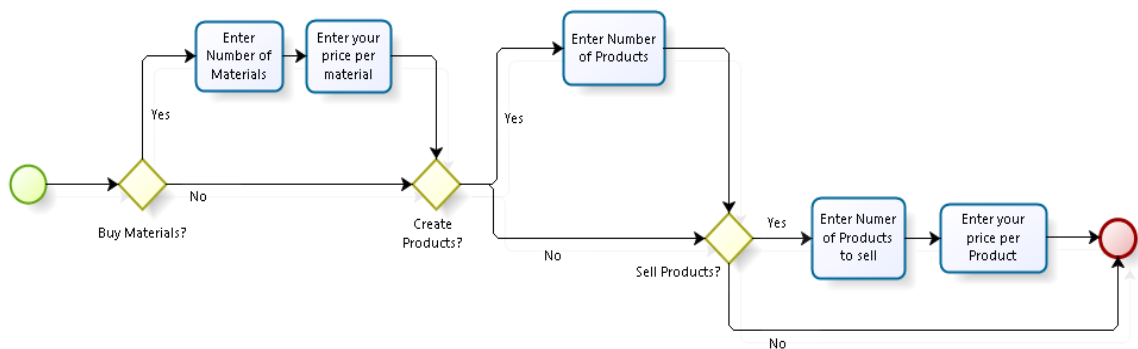


Figure 31 - FactOrEasy Model 1 (partial) [author]

The designed *FactOrEasy Model 2 (partial)* looks like below:

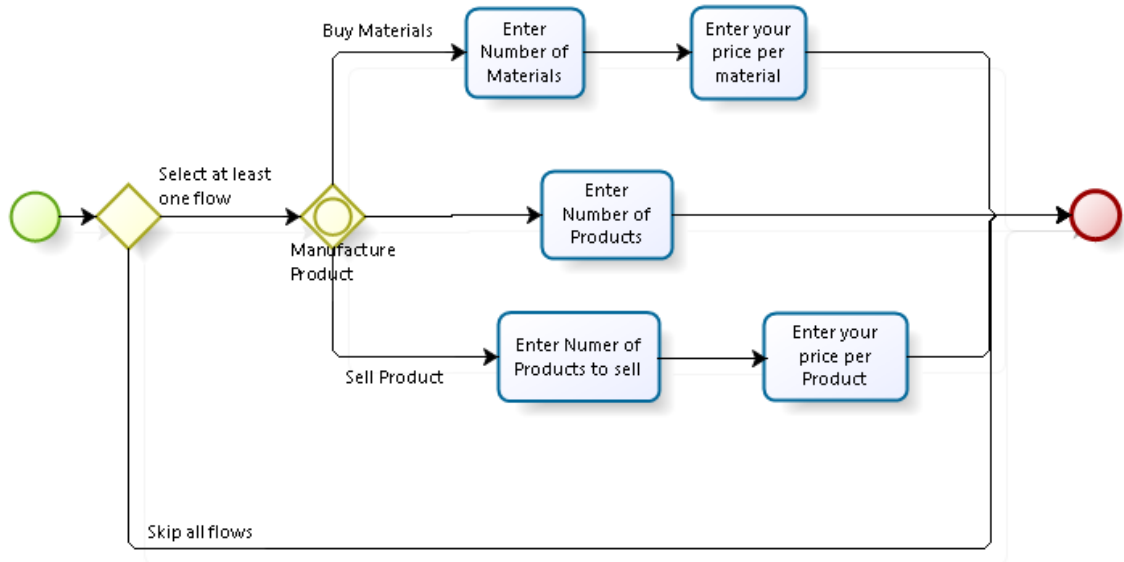


Figure 32 - FactOrEasy Model 2 (partial) [author]

Intentionally we have made both models with the 5 activities so there isn't much difference when it comes to the visual complexity of the model. The aim of this experiment is to prove that *Compound Complexity (BPM)* metric is sensitive enough to estimate complexity proportionally with the user's comprehension and understanding of the model through the BPMN diagram.

Since now we have a foundation of our models, we can continue to add extra elements to it to correctly model the whole game process. In the two models displayed above we have described just one round of play.

We mentioned that the player can take a loan or purchase a factory anytime during the game process so we have to add these two activities preceded by an OR gateway (in Model 2, we just add an extra outgoing path in the already existing OR gateway). Even though these two activities are independent of the 3 flows mentioned in the beginning of this section, loan is dependent on the periodic payments after each round so the *Loan* activity has to enter the AND event which is triggered by the end of the round.

Factory Purchase activity on the other hand is not dependent on rounds of the game so it can skip the AND event gateway.

To continue the game we replace the END event with an AND event that will simultaneously trigger two activities. First activity will calculate the player's cash balance of the previous round with the following logic:

$$\text{Cash Balance} += \text{Selling Price} * \text{Units Sold} - (\text{Factory Costs} + \text{Material Storage Costs} + \text{Product Storage Costs} + \text{Periodic Payment})$$

The second activity will deduct the roundly periodic payment to the existing loan (if there is no loan, the value will simply be displayed as 0)

$$\text{Loan Balance} -= \text{Periodic Payment}$$

After these two activities are displayed they will be collapsed into one process flow again and the next decision flow will be to check if *Cash Balance* is positive. If not, the player has bankrupted and the game is lost. This is modelled by a *Bankrupt* activity and an END event.

If *Cash Balance* is positive, there is the next decision flow which will check if number of rounds is 12. If not, the process flow will go back in the beginning while incrementing the number of rounds by 1, similarly like a *for* loop in a software algorithm.

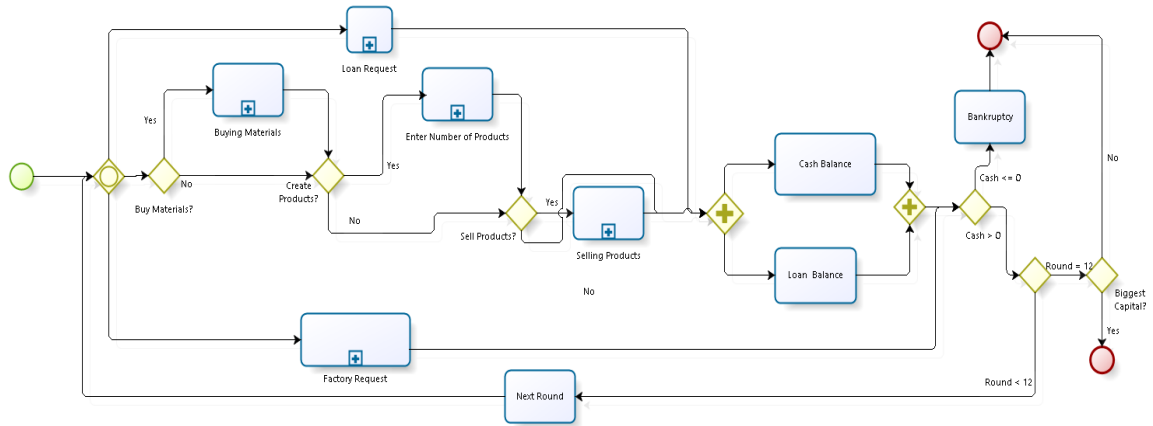
If the number of rounds is 12, it signalizes that it is the end of the game and there will be a last decision flow checking if the player has the biggest capital compared to the other players. If not, the player has lost the game and the process flow connects to the END event that was defined for bankruptcy. If however, the player has the biggest capital from the other competitors, he will win the game and the process flow will go to a new END event which will imply that the game is won.

One thing to consider is that since this process grew in complexity, the initial 3 basic flows (Buy, Produce, and Sell) will be transformed to sub-processes⁶, partially because it will help reduce the number of activities and mostly because each of these process flows have functionalities inside so it makes sense to consider them as sub-processes.

⁶ The 3 diagrams of these sub-processes are available in the Appendix 1

So now, after we have described in few paragraphs how the business process model will look, we can design the full BPMN diagram for both models.

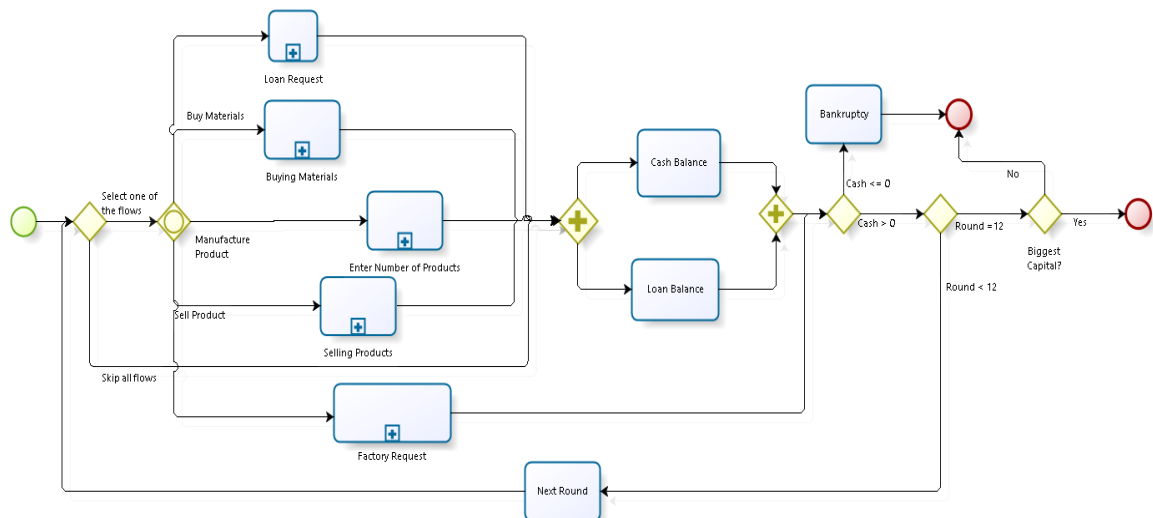
The first model looks like below:



Powered by
bizagi
Modeler

Figure 33 - FactOrEasy Model 1 (full) [author]

And the second model looks like below:



Powered by
bizagi
Modeler

Figure 34 - FactOrEasy Model 2 (full) [author]

Two business process models designed above perform similar flows and have the exact same number of activities however they differ on the type of gateways used. Based on the visual comprehension only, *Model 2* looks slightly less overwhelming and easier to follow than *Model 1* but let's see how these diagrams are evaluated by our developed metric and other existing complexity metrics.

8.3. Model Evaluation by Compound Complexity

After we have designed our two models, let's see how Compound Complexity (BPM) values their complexity.

The specifications for the first model, *FactOrEasy Model 1* are:

FactOrEasy Model 1	
Number of Activities	9

Element Type	Element Count	Cognitive Weight	Element Value
Start Event	1	2	2
End Event	2	2	4
XOR Gateway (n=2)	6	2	12
AND Gateway (n=2)	2	2	4
OR Gateway (n=3)	1	5	5
Model Value (p)			27

Table 3- *FactOrEasy Model 1* element specifications [author]

Now, we can calculate the value of Compound Complexity (BPM) for *Model 1*

$$CC_{BPM}(Model\ 1) = 9^2 + 27 \ln(27) = 81 + 89 = 170$$

The specifications for the second model, *FactOrEasy Model 2* are:

FactOrEasy Model 2	
Number of Activities	9

Element Type	Element Count	Cognitive Weight	Element Value
Start Event	1	2	2
End Event	2	2	4
XOR Gateway (n=2)	4	2	8
AND Gateway (n=2)	2	2	4
OR Gateway (n=5)	1	5	5
Model Value (p)			23

Table 4 FactOrEasy Model 2 element specifications [author]

The value of Compound Complexity (BPM) for *Model 2*

$$CC_{BPM}(Model\ 2) = 9^2 + 23 \ln(23) = 81 + 72.1 = 153.1$$

Based on the Compound Complexity (BPM) metric, *Model 2* is showing to be slightly less complex than the *Model 1*. This confirms our visual observations we made in the previous section. Although the difference in complexity values is subtle, we can agree that this is a reflection of the similarity in the difficulty level of comprehending both models visually.

8.4. Model Evaluation by Existing Quality Metrics

Now, let's see how the complexity level of these two business process models is measured by the existing quality metrics. This evaluation will give us a better idea where our *Compound Complexity (BPM)* metric stands compared to the established metrics which are used in the business process modelling standard.

To measure models with existing complexity metrics we will use a metrics calculation software which is running in Czech University of Life Sciences (CULS) application server⁷. This complexity metrics software is based on the diploma research of Richard Mach (41) leaded by the supervisor of this thesis, J. Pavlicek. It calculates complexity values based on XPDL file standard and is free to use for anybody.

This web application calculates different ranges of quality metrics for business process models based on:

⁷ <http://athena.pef.czu.cz:8080/BpmMeasuresWebClient/>

Size - Number of Activities (NOA), Number of Activities and Control-flows (NOAC)

Modularity – Fan In/Fan Out modularization

Structure - Interface Complexity (IC), Maximum Nesting Depth (MaxND),
Mean Nesting Depth (MeanND), Cyclomatic Number (CN)

Complexity – Control-flow Complexity (CFC)

Comprehensiveness – Coefficient of Network Complexity (CNC)

These quality metrics are the most used in the field of business process modelling (26) and they cover a lot of factors when it comes to evaluating model clarity, effectiveness and comprehension.

To design our case study FactOrEasy models we have used Bizagi Modeler which supports BPMN notation, so diagrams can easily be exported as XPDL files. To measure our models, we upload them in the web application as XPDL files and they are instantly calculated. We have to take into consideration that our models will be calculated with full expanded sub-processes but since we have the exact same sub-process design for both models, the potential increase in complexity will be proportional.

Results for *Model 1* are shown in the following format:



Figure 35 - FactOrEasy Model 1 evaluation by BPM Measures Web Client [author]

Results for *Model 2* are shown in the following format:



Figure 36 - FactOrEasy Model 2 evaluation by BPM Measures Web Client [author]

Since the format of the results is not very comprehensible, we will analyze these results in the following table, to have a clearer picture of how did our two models do when measured by existing BPM metrics.

For the sake of correctness, we will remove from consideration the following metrics:

Maximum Nesting Depth, Mean Nesting Depth – they are irrelevant to the nature of our model, they show same values for both Model 1 and Model 2 (MaxND = 1, MeanND = 0.166)

Fan In/Fan Out – null values

Number of Activities – it is clear from the beginning of the process models' design that we want number of activities to be the same, in order to remove differences between their sizes.

The values calculated from the complexity metrics software in regard to *Model 1* and *Model 2* are shown in the table below. In the last row we have added our proposed Compound Complexity (BPM) metric to see how it compares in model measurement with the existing metrics.

The main goal of this measurement is to see if our hypothesis that Model 2 is less complex than Model 1 is supported by existing quality metrics and what is the differential in complexity between these two models.

Table 5 - Quality metrics measurement on Model 1 and Model 2 [author]

Quality Metrics	M1	M2	M2 < M1	Difference in Complexity
NOAC	55	53	Yes	3.7 %
CN	51	49	Yes	4 %
CNC	2.56	2.5	Yes	2.4 %
CFC	39	59	No	-51 %
IC	292	287	Yes	2 %
CC _{BPM}	170	153.1	Yes	10 %

8.5. Evaluation of Results

From these results, 4 out of 5 selected existing metrics value Model 2 to have less complexity compared to Model 1, hence supporting the results of Compound Complexity (BPM). For these 4 metrics, Model 2 averages 3% less complexity compared to Model 1 while Compound Complexity (BPM) values Model 2 to be 10% less complex than Model 1. This is a very close estimation, considering that each of the metrics represent a different quality perspective in a model.

A clear outlier is shown to be Control-flow Complexity (CFC) which is the only metric that values Model 2 to be more complex than Model 1 by a huge difference of 51%. This is because CFC for every additional outgoing path out of an OR-gateway, increases its complexity value exponentially thus making it heavily biased against any model that uses the OR-gateway which is the case with Model 2.

This observation shows that the our developed metric, Compound Complexity (BPM) has good characteristics when it comes to detect not just the quality of the process model but generally it has inclination to support the model that has a better, simpler and clearer design.

9. Conclusion and Future Work

The main goals of this diploma thesis were to analyze the current state of the quality metrics in the field of business process modelling and based on the more recent researches, try to propose a new quality metric. This thesis has fulfilled its specified goals.

A similarity in nature between software development and business process models has concentrated the effort to adjust complexity measures in software design for use in BPMs. From the literature review conducted, there were 22 existing quality metrics for business process models. However, since this is a relatively new field there are still no fine standards that determine what constitutes a good process model so we can't prove that these existing measures reflect all the attributes of a business process model's clarity and understanding.

The proposed Compound Complexity (BPM) metric that was developed in the practical part of this diploma thesis showed promising results in its ability to measure different aspects of quality within a business process model. This metric was developed by combining the works of Metcalfe on network value, Keating on design partitioning and the rendering of these researches in defining a complexity metric for domain models in a recent publication (2015) by Stuikys et al. We found this metric very useful since domain models, like business process models use similar flow objects such as AND, XOR, OR.

Although initially based on Feature Diagrams (FD) Compound Complexity, we have redefined the formula to better measure the "value" of BPMs by introducing $p \ln(p)$ as a better estimate of complexity growth compared to quadratic growth (Metcalfe) or FD Compound Complexity value (Stuikys et al.). Using a new approach towards the complexity measurement based on a recent research paper (Odlyzko et al.), it was stated that natural logarithm best measures the increase in network's complexity. Once we translated this concept to BPM, we found that growth by natural logarithm really balances the distribution of values between number of activities and cognitive weights of flow objects so we redesigned the second part of the Compound Complexity (BPM) equation to reflect this change and initial results were improved vastly.

By constructing the metric in such a way that it measures both the size (number of activities) and the interface complexity (cognitive weight) as two of the most important factors that enable users to understand business process models, we have attempted to quantify the visual comprehension as much as the mathematically described attributes.

As a case study, we designed two business process models for a real life application in order to test how well our metric is doing in terms of detecting model clarity, simplicity and completeness. The application used in the case study was a factory management game called FactOrEasy, a game intended for management students. The results showed that not only that Compound Complexity (BPM) can detect simpler models but also shares very similar results to different existing quality metrics that measure specific attributes like structure, size or comprehensiveness complexity.

Although the presented case study supports theoretical assumptions, in order for this metric to reach a mature level on measuring business process models, more empirical research and metric testing in a various sorts of models is needed. This is going to be part of the future work in which we will try to publish a research paper in one of the credited scientific journals.

10. References

1. REIJERS, H. A. *Implementing BPM systems: the role of process orientation*. s.l. : Business Process Management Journal, 2006, Vol. 12, pp. 389– 409.
2. MENDLING, J., REIJERS, H. A. and CARDOSO, J. What makes Process Models Understandable? *3rd International Conference, BPM 2005*,. 2005, pp. 48-49.
3. DIJKMAN, R., et al. Similarity of business process models: Metrics and evaluation. *Information Systems*. April 2011, Vol. 36, 2, pp. 498-516.
4. HAMMER, M. and J. CHAMPY. Reengineering the corporation: A manifesto for business revolution. *Business Horizons*. 1993, Vol. 36, 5, pp. 90-91.
5. GUHA, S. and W.J. KETTINGER. Business Process Reengineering. *Information Systems Management*. 1993, Vol. 10, 3, pp. 13- 22.
6. KOHLI, R. and S. SHERER. Measuring Payoff of Information Technology Investments: Research Issues and Guidelines. *Communications of the Association for Information Systems*. 2002, Vol. 9, 14, pp. 241-268.
7. ANAND A., et al. A Literature Review on Business Process Management, Business Process Reengineering, and Business Process Innovation. *The 9th International Workshop on Enterprise & Organizational Modelling and Simulation (EOMAS 2013)*. 2013.
8. DEVARAJ, S. and R. KOHLI. Performance Impacts of Information Technology: Is Actual Usage the Missing Link? *Management Science*. 2003, Vol. 49, 3, pp. 273-289.
9. HUANG, Z., et al. Reinforcement learning based resource allocation in business process management. *Data & Knowledge Engineering*. 2011, Vol. 70, 1, pp. 127-145.
10. MINHONG W., and W. HUIQING. From process logic to business logic—A cognitive approach to business process management. *Information and Management*. 2006, Vol. 43, 2, pp. 179-183.
11. WESKE, M., W.M.P. van der AALST, and H.M.W. VERBEEK. Advances in business process management. *Data & Knowledge Engineering*. 2004, Vol. 50, 1, pp. 1-8.

12. HAMMER, M., and J.CHAMPY. Reengineering the corporation: A manifesto for business revolution. *Business Horizons*. 1993, Vol. 36, 5, pp. 90-91.
13. MIAO Y. How Does the Enterprise Implement Business Process Reengineering Management. *E-Business and E-Government (ICEE), 2010 International Conference*. 2010, pp. 4100-4102.
14. CHEN, Y.C. Empirical Modelling for Participative Business Process Reengineering. *PhD diss., University of Warwick*. 2001.
15. SMITH, H. *P-TRIZ in the History of Business Process*. s.l. : BPTrends, BPTrends, 2006.
16. SCHUMPETER, J.A. The Theory of Economic Development. *Harvard University Press*. 1938.
17. ROGERS, E. *Diffusion of Innovations*. New York : Free Press, 1995.
18. DEWAR, R.D.D., and Jane E. The adoption of radical and incremental innovations: an empirical analysis. *Management Science*. 1986, Vol. 32, 11, pp. 1422-33.
19. SERRANO, A. and M. den HANGST. Modelling the integration of BP and IT using business process simulation. *Journal of Enterprise Information Management*. 2005, Vol. 18, 5/6, pp. 740-759.
20. McAFEE, A. and E. BRYJNJOLFSSON. Investing in the IT That Makes a Competitive Difference. *Harvard Business Review*. 2008, July-August.
21. SCHOLZ-REITER, B., and E.STICKEL. *Business Process Modelling*. 1996.
22. LAMPATHAKI F., KOUSSURIS S., PSARRAS J. *Business Process Modelling - Business Process Engineering*. s.l. : National Technical University of Athens, 2013.
23. Bizagi Process Modeler. <http://www.bizagi.com>. [Online] [Cited: 25 January 2016.] <http://www.bizagi.com/eng/downloads/BPMNbyExample.pdf>.
24. Object Management Group. <http://www.omg.org/spec/BPMN/2.0/>. [Online] [Cited: 16 March 2016.]

25. CARDOSO, J. How to measure the control-flow complexity of web processes . *The Workflow Handbook*. 2005, pp. 199-212.
26. PAVLICEK J., HRONZA R., MACH R., NAPLAVA P.,. Measures of Quality in Business Process Modelling. *CTU Faculty of Information Technology*. 2015.
27. McCABE, T.,. A Complexity Measure. *IEEE Transactions of Software Engineering*. 1976, Vols. SE-2, 4, pp. 308-320.
28. *A Discourse on Complexity of Process Models (Survey Paper)*. CARDOSO, J., et al. Vienna : Business Process Management Workshops. pp. 117-128. September 2006.
29. LATVA-KOIVISTO, Antti M. *Finding a complexity for business process models*. s.l. : Helsinki University of Technology, February 2001.
30. STUIKYS, V., and R. DAMASEVICIUS. Measuring Complexity of Domain Models by Feature Diagrams. *Information Technology and Control*. 2015, Vol. 38, 3.
31. Institute, Software Engineering. <http://www.sei.cmu.edu/productlines/>. [Online] [Cited: 22 February 2016.] <http://www.sei.cmu.edu/productlines/>.
32. KEATING, M. Measuring Design Quality by Measuring Design Complexity. *Proceedings of the 1st Intl. Symposium on Quality of Electronic Design (ISQED 2000)*. pp. 103-108.
33. J., SHAO J. and WANG. A New Measure of Software Complexity based on Cognitive Weights. *Canadian Journal of Electrical and Computer Engineering*. 2003, Vol. 28, 2, pp. 69-74.
34. LI, G. Economic sense of Metcalfe's Law. *Proceedings of 17th Intl. World Wide Web Conference*. 2008, pp. 103-108.
35. WANG. Y. On cognitive informatics: Keynote lecture. *Cognitive Informatics (ICCI'02)*. 2002, pp. 34-42.
36. HOARE, C.A.R., et al. Laws of Programming. *Comm. ACM*. Vol. 30, 8, pp. 672-686.
37. LAUE, V. and R. GRUHN. Adopting the Cognitive Complexity. *5th IEEE Int. Conf. Cogn. Informatics*. 2006, Vol. 1, pp. 236-241.

38. ODLYZKO, A., TILLY, B., and BRISCOE B.
<http://spectrum.ieee.org/computing/networks/metcalfes-law-is-wrong>. *IEEE Spectrum*.
[Online] [Cited: 12 March 2016.]
39. ODLYZKO, Andrew, and Benjamin TILLY. A refutation of Metcalfe's law and a better estimate for the value of networks and network interconnections. *Manuscript*. 2 March 2005.
40. <http://factoreasy.cz/FactOrEasy/>. [Online] [Cited: 13 March 2016.]
41. Mach, Richard. Návrh a tvorba nástroje pro optimalizaci procesů na základě analýzy BPM modelů. *Fakulta informacních technologií, CVUT*. 2015.

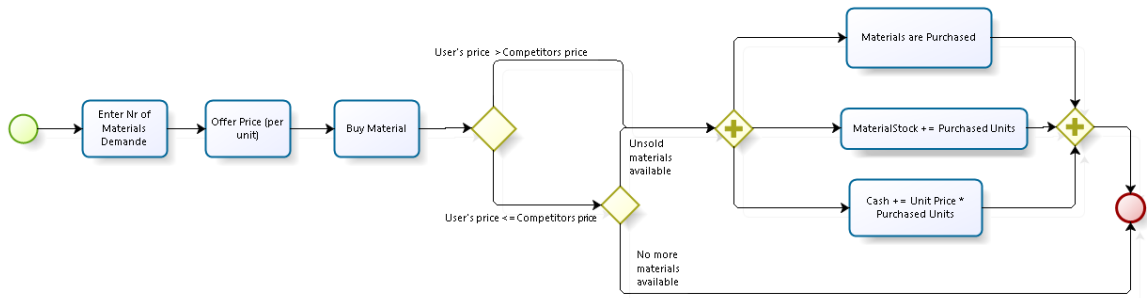
11. Table of Figures

Figure 1 - Typology of Business Processes	10
Figure 2 - The evolution of BPM [source: http://dashflows.com/workflows]	12
Figure 3 - Interaction Areas of Business Process Modelling [author].....	16
Figure 4 - A BPM Hourglass	20
Figure 5 - Events [author]	21
Figure 6 - Activities [author]	21
Figure 7 - Gateways [author].....	22
Figure 8 - Connectors [author].....	22
Figure 9 - Swimlanes [author]	23
Figure 10 - Artifacts [author].....	23
Figure 11 -Credit Application Process [author].....	25
Figure 12 - Check Applicant Information Sub-Process [author].....	25
Figure 13 - Credit Application Process with added sub-process [author]	26
Figure 14 - Credit Application Process with event based gateways [author].....	27
Figure 15 - Credit Application Process - final [author].....	28
Figure 16 - Frequency of BPM metrics determined by CTU team (26).....	30
Figure 17 - XOR-split CFC [author using Bizagi Modeller].....	31
Figure 18 - OR-split CFC [author using Bizagi Modeller].....	31
Figure 19 - AND-split CFC [author using Bizagi modeller]	32
Figure 20 - Credit Application sub-process with added legend for CNC [author].....	34
Figure 21 - A feature diagram of the GATE domain model [source: Stuikys, et al. (30)] ..	38
Figure 22 - Cognitive weights of Feature Diagram elements [source: Stuikys, et al. (30)]	40
Figure 23 - GATE Feature Model with explained relationship types [author].....	43
Figure 24 - Cognitive weights for BPMN elements [author]	49
Figure 25 - An OR flow with n=3 [author].....	50
Figure 26 - A design partitioning model mirrored by BPMN [author]	52
Figure 27 - A simple business process model [author].....	54
Figure 28 - A more complex business process model [author]	55
Figure 29 - Complexity growth between functions [author]	57

Figure 30- A screenshot of FactOrEasy game [source: TACR Program Omega].....	61
Figure 31 - FactOrEasy Model 1 (partial) [author].....	63
Figure 32 - FactOrEasy Model 2 (partial) [author].....	64
Figure 33 - FactOrEasy Model 1 (full) [author]	66
Figure 34 - FactOrEasy Model 2 (full) [author]	66
Figure 35 - FactOrEasy Model 1 evaluation by BPM Measures Web Client [author].....	69
Figure 36 - FactOrEasy Model 2 evaluation by BPM Measures Web Client [author].....	70

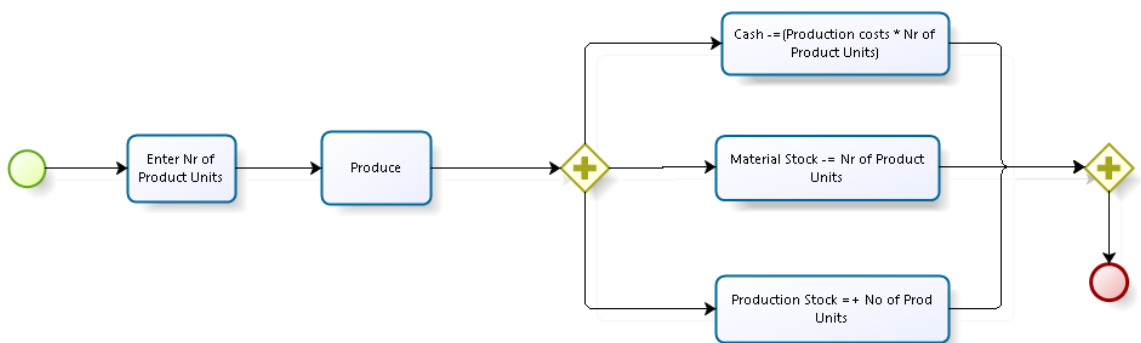
12. Appendix A – Sub-processes for Models 1 & 2

Buying Materials sub-process [author]



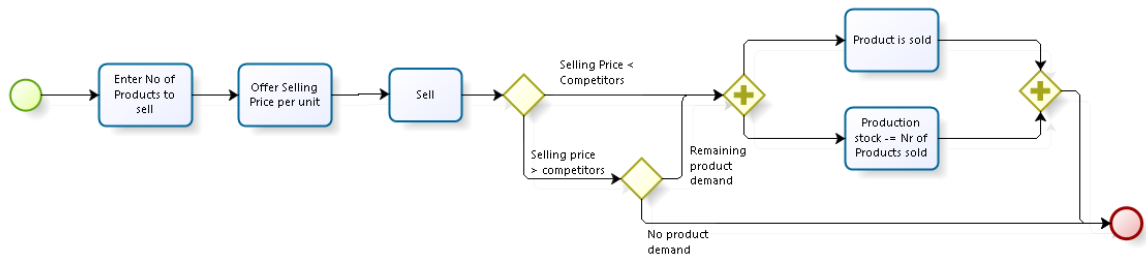
Powered by
bizagi
Modeler

Enter Number of Products sub-process [author]



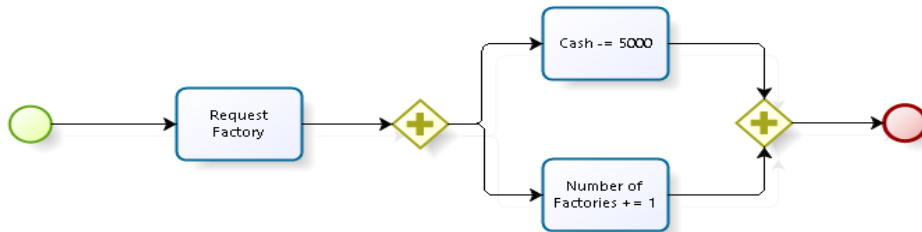
Powered by
bizagi
Modeler

Sell Products sub-process [author]



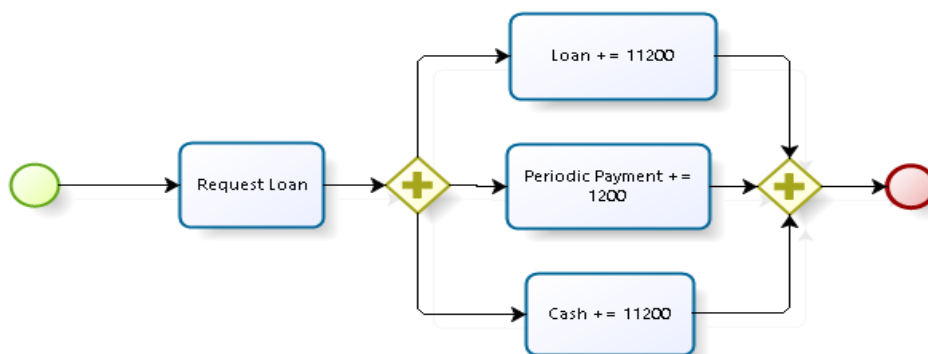
Powered by
bizagi
Modeler

Factory Request sub-process [author]



Powered by
bizagi
Modeler

Loan Request sub-process [author]



Powered by
bizagi
Modeler