UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Rando Tõnisson

Comparison of Business Motivation Model and Intentional Distribution

Bachelor's Thesis (9 ECTS)

Supervisor: Raimundas Matulevičius

Tartu 2016

Comparison of Business Motivation Model and Intentional Distribution

Abstract:

Goal modelling is an important activity to reason why different software decisions are taken, or architecture solutions are implemented. Currently there exist a number of goal-oriented modelling approaches. In this thesis, the semiotic quality framework is applied to compare the quality of the business motivation model (BMM) and Intentional Distribution (*i**) modelling languages at the coarse-grained level. The thesis reports on the BMM and *i** language quality and model quality. The thesis also presents observations on how the BMM and *i** modelling languages could be used to reason on and support construction of the business process models expressed in business processes model and notation.

Keywords:

BMM, the i* framework, SEQUAL, business process management, language quality, model quality

CERCS: T120: Systems engineering, computer technology

Business Motivation Model'i ja *Intentional Distribution*'i võrdlus

Lühikokkuvõte:

Eesmärkide modelleerimine on tähtis, et mõista erinevate tarkvara või arhitektuuri valikute tagamaid. Praegu leidub mitmeid eesmärkidele suunatud modelleerimise võtteid. Selles töös on kasutatud *semiotic quality framework*'i, et võrrelda *Business Motivation Model*'i (BMM) ja *Intentional Distribution*'i (*i**) kvaliteeti üldistatud tasandil. Töös hinnatakse *BMM*-i ja *i**-i keele kvaliteeti ja mudeli kvaliteeti. Lisaks kirjeldatakse kuidas *BMM*-i ja *i**-i keeli saab kasutada äri protsessi mudeli, siin *business processes model and notation*, tegemisel.

Võtmesõnad:

BMM, *i**, SEQUAL, äri protsesside juhtimine, keele kvaliteet, mudeli kvaliteet

CERCS: T120: Süsteemitehnoloogia, arvutitehnoloogia

Contents

1 Introduction

With the technological evolution, the demand for requirements engineering (RE) is growing. Goal-oriented modelling is one of the most important research developments in the RE field. Traditional analysis focuses on answering the questions *what* and *how*, where goal-oriented analysis has shifted its focus to *who* and *why*. As the importance of quality in the analysis in business process management is increasing, it is necessary to understand the rationale behind the changes in the business process or software system. The best way to cover all this is to start using a combination of goal-oriented approaches and business process modelling.

One of the options is to use business motivation model (BMM) that supports the identification of the motivation behind the change and define the means to make the changes. It also shows the relationships between all the components needed for carrying out the changes (Object Management Group, 2015). The second option is to use a traditional goal-oriented modelling for early stages of RE. In addition to that, goal-oriented modelling can support and reason the decisions taken during the business process management. One of the most widely used frameworks (Ayala et al., 2005) for this kind of goal-oriented modelling is the *i** framework. It supports the understanding of the actors, their goals and the means available to achieve their goals (Yu, 1995).

In this thesis, the semiotic quality framework (SEQUAL) (Krogstie, 2012) is used to compare both the modelling languages *i.e.*, the BMM and *i**. Three research questions are analysed:

RQ1. Which modelling language – BMM or *i** – is of better quality?

RQ2. Which model – BMM or *i** model – is of better quality?

RQ3. How BMM and *i** models could provide rationale for business process modelling?

To answer those research questions, an overview of BMM and *i** is provided that consists of an overall overview, semantics, abstract syntax, and an example. The example is made to cover most of the concepts in the modelling language used. Using both of the examples, a BPMN (Business Process Model and Notation) model is made, while giving an overview of the process of making it. For the comparison, overview of the framework (SEQUAL) is given. Also, the criteria for the comparison is defined for language quality evaluation and model quality evaluation. Using that criteria, the evaluation and comparison is made. Results of this thesis are submitted to the international conference (Tõnisson and Matulevičius, 2016).

The thesis is structured as follows: Chapter 2 has the overview of BMM and the modelling example, Chapter 3 has the overview of *i** and the modelling examples. In Chapter 4, the BPMN model based on the previous models is described. After that, in Chapter 5, overview of the SEQUAL is presented, with the criteria for later evaluation. The criteria defined in Chapter 5 is used in Chapter 6, where the evaluation and comparison is made. Finally, in Chapter 7, the thesis is concluded.

2 Business Motivation Model

Business Motivation Model (BMM), is developed by the Business Rules Group (BRG). The first edition, version 1.0, was published on the BRG website (www.businessrulesgroup.org) in November 2000. Currently the latest version of BMM is 1.3, published in September 2007. All of the information is based on BMM version 1.3 documentation (Object Management Group, 2015).

The BMM provides an organized structure for developing, communicating and managing business plans. Specifically, the BMM identifies factors that motivate the making of business plans, identifies and defines the elements of business plans and shows how the elements relate to each other. Among those elements are Business Policies and Business Rules, which provide governance guidance for the business (Object Management Group, 2015).

2.1 Overview of BMM

The BMM contains a set of built-in concepts, which define the elements of business plans. They are joined in a structure that will support a multiple approaches to create and maintain a BMM and is made to support the changing within the business itself. The BMM also contains the structure for external elements. They are defined as: Business Process, Business Rule, and Organization Unit. They are needed in the BMM to govern the process of changing. In the BMM the external elements are not defined, but will be references to objects outside the BMM itself.

The BMM concepts are developed to be very simple to understand, which means those concepts only have identifiers and text descriptions. Most of the associations between the objects are optional to have and many-to-many.

According to the creators of BMM, it is not intended to be used as a full business model or the specification for it. It is also not made to support development or project management process or tool. It still could be used to do those activities, but may cause unexpected errors in the BMM.

All the elements of BMM are developed with the business in mind. It contains two major areas: the Ends and Means and also Influencers and Assessments of the business plans.

The Ends is the goal that the business wants to succeed in and the Means are the processes that are employed to get to the End and. Influencers are the cause to do something in the business as they provide the motivation to achieve the goals. They shape the elements in the business plan and also are the base for Assessments that impact the Ends and Means. All of those are related to each other by some fundamental questions, which the BMM will need to answer. The questions are “What is needed to achieve in order to achieve what the business wants to achieve?” and “Why does each element of the business plan exist?” (Object Management Group, 2015). The first one is answered by completing the Means and Ends part of the BMM and the second is answered by analysing the Ends and Means to get the motivation to work to achieve the end goals.

2.2 Semantics of BMM

The most important part of BMM is the motivation, an enterprise should not act randomly. To do something the business should have a detailed answer to the question "Why it should be done?", and this is the motivation for BMM.

The motivation to do something can come from inside the business from the authority or outside of the business. In both cases, it should be clear what the end goal is and why it is that. For the authority they are most likely to set up the goals. For most cases the authority wants changes to get the best for the business and doing that it should be clear who and why the influencer is that important. In practise, businesses do not have a clear traceability of motivation to move on, so the BMM will provide support for it.

BMM has a fundamental assumption that the enterprise is not driven by change but by how it decides to react to the change. It is important to monitor the influencers to react to the change and also make the assessments on their impact to the enterprise. Whatever process to react to the change is used, the model supports traceability to forward and also backward. Both are very important for regulatory compliance.

The BMM is made to be as simple as possible, one effect on this is the separation of concerns, the distinction between Ends and Means, Influencers, Assessments, Courses of Action and Business Policies.

The BMM can be applied to any business large enough to need to define and manage their own business plans. This could mean that smaller units in the same bigger enterprise have their own BMM. That kind of BMM most likely has objects owned and defined by the bigger.

The three decomposition in the BMM are desired result, course of action and business policy. In practice, the boundaries between those are defined by the organization unit, to match the structure set in the enterprise itself.

2.3 Abstract syntax of BMM

The syntax is illustrated by a class diagram (Figure 1). The class diagram is made by using the BMM documentation (Object Management Group, 2015). The words in *italic* represent the classes in the diagram.

End is something that the enterprise wants to be. It can be something that the enterprise will want to be or even just maintaining its market and competition place. By definition, the end will not contain the information on how it should be achieved. In BMM the *End* is divided into *vision*, *goals* and *objectives*, where the last two make up the *desired results.* The *vision* is an overall image of the desired end. It is possible to use the BMM without defining the *vision* in depth. The *goals* are more long term and is defined more qualitatively and narrowly. This is needed so *objectives* could be defined for it. The *objective* will provide information on the progress being done towards achieving the *goals*.

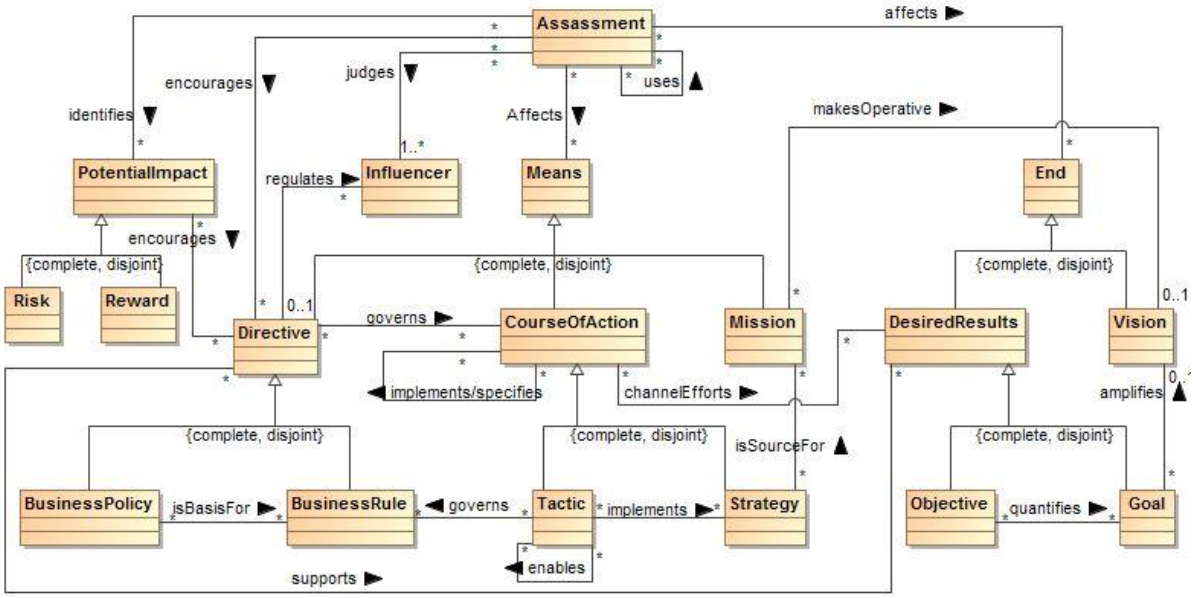

Figure 1. – Syntax of BMM, adapted from (Object Management Group, 2015).

Means are the steps that the enterprise has decided to take to get where it wants to be. A mean can be a device, technique, instrument, method, etc. that can be called upon or activated to achieve the *end* by being the source for *mission and* channelling the efforts towards *desired results*. It does not include business process nor workflow necessary to carry out such tasks.

In BMM *means* are divided into *mission*, *courses of action*, and *directives*. A *mission* is an ongoing activity in the enterprise and should cover all the *strategies* and *tactics*. BMM can be used without defining a *Mission* explicitly.

The *course of action* is something that the business has decided to do and how well it must be done and is more than just a simple resource or skill that an enterprise can use. It is a way to configure the aspects of the enterprise like processes, locations, and people to get to the *desired results*. It is a result, which the enterprise has chosen as the best way to use its resources.

In the BMM, the *course of action* is categorized as *strategies* and *tactics*. The differences in those two are vague in the models definition, but the user can make their own criteria to difference between those two. By default the *tactic* is something that implements *strategy*.

Strategies are normally long term and fairly broad in scope. *Strategies* are implemented by t*actics*, which are shorter term and more narrow in the scope. A single *tactic* may be used with multiple *strategies*. Generally, *strategies* are selected to get to the *goals* and *tactics* are used to meet the *objectives*. It is not required to follow and enterprise can change it as they seem to need.

The d*irectives* are categorized as *business policies* and *business rules*. In general, *business policies* exist to control and guide the s*trategies* and *tactics*. They define the limits on how and when something can or cannot be done. Compared to *business rule*, the *business policy* can be less formally-structured and may be less formal and broader in general. *Business policies* are not directly practicable and so practiced *directives* are *business rules*. So the *business rules* are

defined as such as they can be used in practise. *Business rules* are the basis for the *business policies,* as they are not directly practicable.

An *influencer* is somebody or something that causes the need to change in an enterprise. For such, it uses the *means* to get to the *end*. Also, it may confirm the need to not change where something was expected, but it did not happened. *Influencers* may be internal or external. For a smaller unit inside a larger organization, the smaller can use the larger as an external *influencer*. In most cases, *influencers* are something that the companies define themselves, also it is possible to use the set provided by the BMM.

A change caused by an *influencer* is neutral. It is in a neutral position until the business decides how to react to it. An *assessment* is the judgment that the business decided how the *influencer* may affect the business and how they should employ their *means* to achieve the *end*. The decisions are reflected on the *ends* and *means*. Different people or even same people at different times may judge the *influencer* differently. To cover that, the model supports a record of the judgements made to see the differences in time and people. It can be used in the future to judge similar influencers or writing reports. The BMM supports SWOT analysis for one of the possible ways to get the assessments done, different people or enterprises may choose different approach.

There are three concepts in the BMM that are defined as an external reference. Those are *organization unit*, *business process*, and *business rule*. They are all needed for a detailed model, but their standards and concepts are external to BMM.

Organization unit has two roles: it participates in the making of the making of the BMM and defines the boundaries of the enterprise being modelled. For the first one, *organization unit* participates in *ends* defining, *means* establishing, *assessments* making, *influencers* recognizing and *strategy* planning. Also may be responsible for *business process*.

Business process is a detailed plan based on the *courses of action*. It provides the steps, sequences, structure, interactions, and connections to events that are part of the process. Like *courses of action*, *business processes* are also governed by *business policies*.

Business rules provide a specific, practicable way to implement *business policies*. Some rules are automated by software, while the others are only used by people. *Business rule* is derived from *business policy*, may guide the policy and may affect *tactics*.

2.4 Example of BMM

The example (Figure 2) is made to cover all the main concepts of the BMM and is not made for any specific company - it is made up scenario. The words in *italic* are concepts, and words or phrases in arial font are taken directly from the example to give a better explanation.

BMM is all about the motivation to do something. For this example (Figure 2), the company's *vision* is to have an online shopping system with delivery (*vision* Online shopping and delivery is working). With the *vision* in place, it is needed to find out, why the *vision* is important and why it should be achieved. To do that, the BMM has the *influencer* concept.

In this example, there are three *influencers*: one external and one internal. The external one is the most important one, as it has the main motivation attached to it (*influencer* Customers are

too lazy and busy to go to the shop themselves). As the description appoints, the online shopping system with delivery is built to make the customers life easier and with that gain more control over the market. The other two *influencers*, that are internal and about the infrastructure of the company, are there to help the company to start making decisions (*influencers* Warehouses need rework; Need an online shopping system). Without them there would not be possibility to start the online shopping.

After the i*nfluencers* are in place, it is needed to judge the *influencers*. To do that, there is the *assessment* concept in the BMM. It uses SWOT methodology to judge the *influencers*. There are a total of five *assessments* made in this example.

The first one is made to judge the customers (*assessment* Customers would like an online shopping system with fast delivery). By using SWOT analysis, it is an opportunity as it gives the business a change to get new clients.

The second *assessment* is made to judge the workhouses (*assessment* not enough workers in the warehouse). It will affect the starting of the delivery system. It is a weakness for the company and by making this assessment it must be removed.

The third *assessment* is also about the workers (*influencer* Warehouses need rework). There is a need for the delivery (*assessment* No delivery for the goods), when the online shopping system with delivery should go live. It is a major weakness and without addressing this, the company will not get to their vision.

The fourth *assessment* is about the warehouses themselves (*influencer* Warehouses need rework). In a normal warehouse the delivery system will not be able to start working. It may be not accessible fast enough or even do not have a place to start packing the goods bought online (*assessment* The warehouses are not suitable for delivery).

The fifth and last *assessment* is about the online system (i*nfluencer* Need an online shopping system). It is probably the biggest weakness (*assessment* There is no online shopping system currently working). It also comes with a risk (*risk* Customers may not like the shopping system built). So when making the decisions, that *risk* must also be covered.

To be able to do something, there needs to be a mission *(mission* Be able to buy groceries anywhere, at any time). This makes the *vision* in the *end* operative and helps to come up with the *tactics* and *strategies* meant to achieve the *goals*.

The *tactics* are made by analysing the *assessments* made to judge the *influencers*. There are a total of four direct *tactics* deriving from *assessments*.

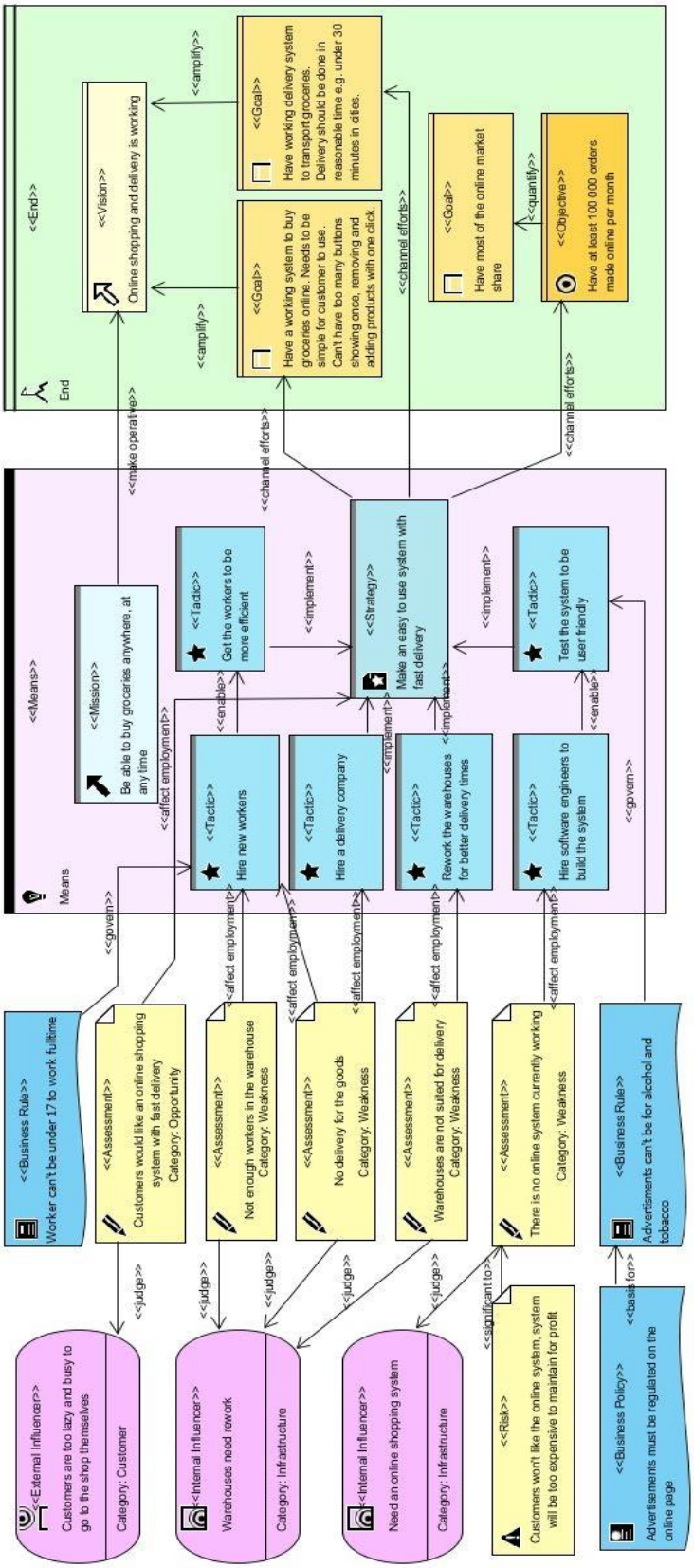

Figure 2. - Example of BMM.

The first *tactic* is a quite vague one (*tactic* Hire new workers). Without the previous judging (*assessment* Not be enough workers in the warehouse) about the warehouse workers, it would be useless as there is no information about the workers needed. It also can be used for delivery aspect, meaning that there is a need for delivery workers. After this *tactic* is in use, it enables another one (*tactic* Get the workers to be more efficient). This *tactic* also has a *business rule* governing it (*business rule* workers cannot be under 17 to work fulltime).

The second *tactic* is gotten directly from the *assessments* (*tactic* hire a delivery company). This one is quite similar to the last one, which was just to hire new workers. It is an alternative route to the same goal, which is to have a delivery system.

The third *tactic* (*tactic* rework the warehouses for better delivery times) derives directly from assessments (*assessment* The warehouses are not suitable for delivery). To follow this tactic, the management would have to find a construction company to do the rework.

The fourth *tactic* will start the online system building (*tactic* Hire software engineers to build the system). The main goal of this *tactic* is to build the online shopping system. After hiring the engineers, it enables the system building and testing. The testers will have to look for major bugs and flaws in the system and also follow a *business rule*. The rule comes from a *business policy (business policy* Advertisements must be regulated on the online page). The *rule* itself is about tobacco and alcohol (*business rule* Advertisements cannot be about alcohol or tobacco).

After the workers are hired and familiar with the workflow and the online shopping system is ready to be used, the *strategy* can be implemented (*strategy* Make an easy to use system with fast delivery). This *strategy* is a direct link from *means* to the *end.*

After the *strategy* is implemented, the *goals* and *objectives* can be achieved. In this case, there are two direct *goals* from the *strategy*, one about the online system and the other one about the delivery system.

The first *goal* is to have a working system to buy groceries online. It also has a more detailed description of the system (*goal* Have a working system to buy groceries online. Needs to be simple for customer to use. Can't have too many buttons showing once, removing and adding products with one click). The long description is there to help to understand the system better.

The second *goal* is to have working delivery system to transport the goods bought online. It also has some more detailed description (*goal* Have working delivery system to transport groceries. Delivery should be done in reasonable time e.g. under 30 minutes in cities).

There is also an *objective* deriving directly from the *strategy* in the means (*objective* Have at least 100 000 orders made online per month).The *objective* quantifies a *goal* (*goal* Have most of the online market share). This goal itself is not required for the vision (*vision* Online shopping and delivery is working) as the *vision* is achieved even if the retail chain does not have the majority of the online shopping shares.

After both the *goals* that amplify the *vision* are achieved, the *vision* itself will be achieved (*vison* Online shopping and delivery is working). This will conclude the BMM.

2.5 Summary

In this chapter the main concepts of BMM are described while giving an overview on the semantics and abstract syntax. For better understanding and also for later evaluation, an example of BMM is made and also described. In the next chapter, discussion on Intentional Distribution (*i**) is given.

3 Intentional Distribution

Intentional Distribution Model (*i**) is a modelling language or framework developed mostly by Eric Yu. In his 1995 paper "Modelling Strategic Relationships for Process Reengineering" he started developing it and was the first time *i** was made known to the public. Yu was concerned about the technology and computers becoming more pervasive and wanted to develop a new and better way to start requirements engineering, especially when there are multiple participants. The participants could be both humans and computers. *i** is made to be simple enough to use without much experience and will still be a good way to show the "work-flow" as a diagram (Yu, 1995).

All of the information is taken from Yu's works, mainly his 1995 paper mentioned earlier (Yu, 1995).

3.1 Overview of *i**

The *i** language is developed to help early stages of requirements engineering. It can be used for a completely new system or process or reengineering an existing one. The *i** model will be a good overview of the requirements needed and the end goals. While making the model, the maker will go over all the participants in the process being modelled and will make out all the dependencies between the participants. Those participants are defined as *actors* in *i**, an *actor* is the central concept in the *i** modelling language.

In *i** *actors* are made to have intentional properties like goals, beliefs, abilities, and commitments. They have their own goals to achieve, tasks to perform and resources to be furnished and depend on each other with them. This dependence allows them to achieve goals that may be difficult or even impossible to achieve on their own. But by depending on each other they also become vulnerable to if some of the actors do not do their tasks (Yu, 1997).

The dependencies may differ from each other. For example, in one case the dependency will involve money or funds, while in the other case one of the actors needs the second one to perform a task of some kind. In *i** the dependencies are categorized by their type. There are a total of four types of dependencies in *i**, which are *resource*, *task*, *goal* and *softgoal dependencies*.

The *i** language consists of two main modelling components, Strategic Dependency (SD) and Strategic Rationale (SR) model. SD is used to describe the relationships of various actors in an organizational context while SR is used to describe the organizations stakeholders' interests and concerns and how they are addressed by the system and environment (Yu, 1997).

3.2 Semantics of *i**

As there are two main modelling concepts in *i**, the semantics part of *i** is divided into two. First part is about the semantics of Strategic Dependency model and the second about the semantics of Strategic Rationale model.

The semantics of Strategic Dependency model. The SD model gives an intentional description of process among the *actors*. Intentional in this case means the capture of underlying motivations behind the activities and flows in a process. The intentional approach lets the *actors* have their own freedom of action but only in the boundaries of social constraints. The model is

rich in concepts and lets the analysts explore a broader implication of processes and activities than more non-intentional models. The model can be basis for stakeholders to analyse their opportunities and vulnerabilities.

The SD model is a network of *dependencies* and relationships between different *actors*. The actors can have multiple *dependencies* from other *actors* and also be depended on by multiple *actors*. One *actor* could also be depended by multiple tasks or resources from the same *actor*.

The SD model aims to capture the *actor's* intentions to do something, instead of the usual non-intentional and non-strategic process models. While doing this, only the essential information is carried over, while the non-essential is leaved out of the model.

The semantics of Strategic Rationale model. The SR model gives a description of tasks and processes in terms of its elements and rationales behind them. While SD gave a more abstract overview, the SR model goes more in depth with the processes and will show the *actors* internal tasks and the reasoning behind those. SR model describes the actors internal relationships, such as *means-ends* relationship, providing explicit representation of the most basic questions "why?" and "how?". Using both of the models together, SR and SD, process alternatives can be found, also *actors* can find new process designs for their interests and needs, which may constantly change.

The SR model is quite similar to the SD model, both have a similar set of modes and links to represent the structure and rationales behind processes. The SR model has *goal*, *task*, *resource*, and *softgoal* as the nodes, which were the types of dependencies for SD model. Those nodes are linked by *task-decomposition* links and *means-ends* links.

Means-ends links in the SR model are seen as application of generic rules in the particular context of the model and *task-decomposition* links are used between the main task and its components. The model elements are included only if they are important enough to influence the achievement of the *goals*.

3.3 Abstract Syntax of *i**

As there are two main modelling concepts in *i**, the abstract syntax part of *i** is divided into two. First part is about the syntax of Strategic Dependency model and the second about the syntax of Strategic Rationale model. The class diagrams (Figure 3; Figure 4) are made by the author and his supervisor and do not have all the classes defined by Yu in his 1995 paper. The classes shown are the ones used in the examples.

The syntax of Strategic Dependency model. The SD model is a set of nodes and links (Figure 3). Each of the node is an *actor* and the links show the *dependencies* between them. The *dependency* shows that one *actor* depend on other to achieve ones *goal*. They both have separate *goals*, but they also may be similar to each other and with that, the *actors* can work together and depend on each other to achieve the *goals*. *Actors* can depend on each other with some object, then the object is called *dependum*. The depending *actor* is *depender* and the actor *dependent* on is called dependee. The *actor* is someone who will use their own powers to achieve their own *goal*, but may depend on other *actors*.

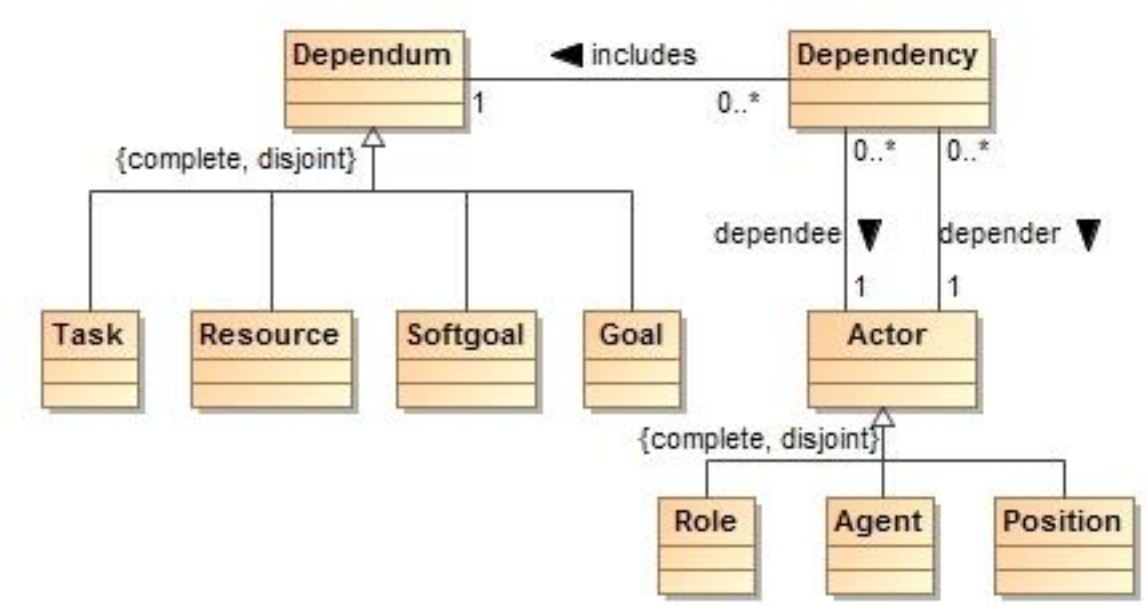

Figure 3. - Syntax of SD model, adapted from (Ayala et al. 2005).

There are four types of *dependencies* in the model, which are distinguished by their type of dependum. Those are r*esource*, *task*, *goal*, and *softgoal dependencies*.

In *goal dependence* a *depender* is depended on the *dependee* by getting the later to achieve something or getting to some point to achieve their own *goal*. The *dependee* is free to achieve it on its own way, all that matters is that the *goal* will be achieved and the *depender* can continue the work. With this *dependency*, the *depender* is given the ability to assume that the *dependees* work will hold, but also a vulnerability, caused by the lack of work by the *dependee*.

The *task dependency* is used when the *depender* is dependent on the *dependee* by some kind of activity, which is carried out by the *dependee*. A task *dependency* will specify how the task is carried out but not why it is done. The *depender* is also vulnerable, as the *dependee* may fail with the *task*.

In *resource dependency*, the *depender* is dependent on the *dependee* for the availability of a physical or informational entity. This dependence grants the *depender* to use the entity as a *resource*, but will be vulnerable for not getting the *resource* from the *dependee*.

In *softgoal dependency*, the first *actor* needs the other actor to carry out some task that meets the *softgoal*. *Softgoal* is something that is required to achieve the main *goals*. Like in *goal dependency*, the *depender* can assume that the *dependees* work condition will stay, but is vulnerable if the condition is not met in the first place.

The *i** language distinguishes among three degrees of *dependency* strength. Those three are *open*, *committed*, and *critical dependency*. For *depender*, a stronger *dependency* means that the *dependee* is more crucial for his work and makes the vulnerability higher. For *dependee* a stronger *dependency* implies that he needs to put more effort to deliver the dependum.

In an *open dependency*, the dependum is not as crucial to the *depender* as in other cases. Missing out on the *task* performed or not having the *resource* available will only affect the *depender* negatively but he could still achieve his *goal*.

In a *committed dependency*, the *depender* has already but a significant effort to achieve his *goal* and missing out on the dependum will make his work harder. The work done cannot be reversed without a loss for the *depender*. For the *dependees* side, he needs to make sure the dependum is delivered and on time for the dependers *goal*.

In critical *dependency, depender's goal* will most likely not be achieved if the *dependum* is not achieved. This means that the *depender* needs to be sure not only about his immediate *dependencies* but also about his *dependees* dependums and also his dependums and so on.

Actors have three categories in the *i** model. Those three categories are based on the social status and help to identify the *actors* better. Those subunits of actors are *agents*, *roles*, and *positions*.

A *role* is a characterization of the behaviour of a social *actor* in a specialized context or domain. Its characteristics can be easily transferred to other *actors* and the *dependencies* associated with the *actor* apply regardless of the player of that *role.*

An *agent* is an *actor*, who has strong characteristics just as a human individual. An *agent* could be a human but also artificial, such as a part of software. That's also why the *agent* is not called a person. The *dependencies* of an *agent* apply regardless what roles the *agent* has. Unlike the *roles, agents'* characteristics are not easily transferable.

A *position* is the intermediate abstraction between a *role* and an *agent*. It is a set of *roles* that are played by one *agent*. The *agent* occupies a *position* and a *position* covers a *role*.

The syntax of Strategic Rationale model. The SR model shows the internal elements of the *actors* (Figure 4). The model gives an overview of the *actors' goals, tasks, resources* and *softgoals*.

A *goal* is something that the *actor* wants to achieve. It could be a specific condition or state of affairs. In the i* language, it is expressed as an assertion. The way of achieving the *goal* is not specified in the model and it will give the opportunity to consider alternatives.

A *task* specifies the way of doing something. When the *task* is a component of a bigger *task,* the smaller will give the restrictions and will make it follow a specific way.

A *resource* is something physical or informational that is not considered a big problem by the *actor*. The only problem concerning it is its availability on the given time and from who it should be gotten if it is not an internal *resource*.

A *softgoal* is a condition in the world, which the *actor* would like to achieve. Unlike the *goal,* a *softgoal* is not as sharply defined and is subject to interpretation. While being a component to a *task,* the *softgoal* will give the required quality to that *task.*

To see how the *task* is performed, it is done by describing the elements or components of the *task*. The elements are linked to the *task* node by *decomposition* links. The *decomposition* links are categorized as *subgoal, subtask, recourceFor,* and *softgoalFor,* which are corresponding to

the four types of nodes. Those can also connect up with SD models when the reasoning goes beyond the *actors* boundaries.

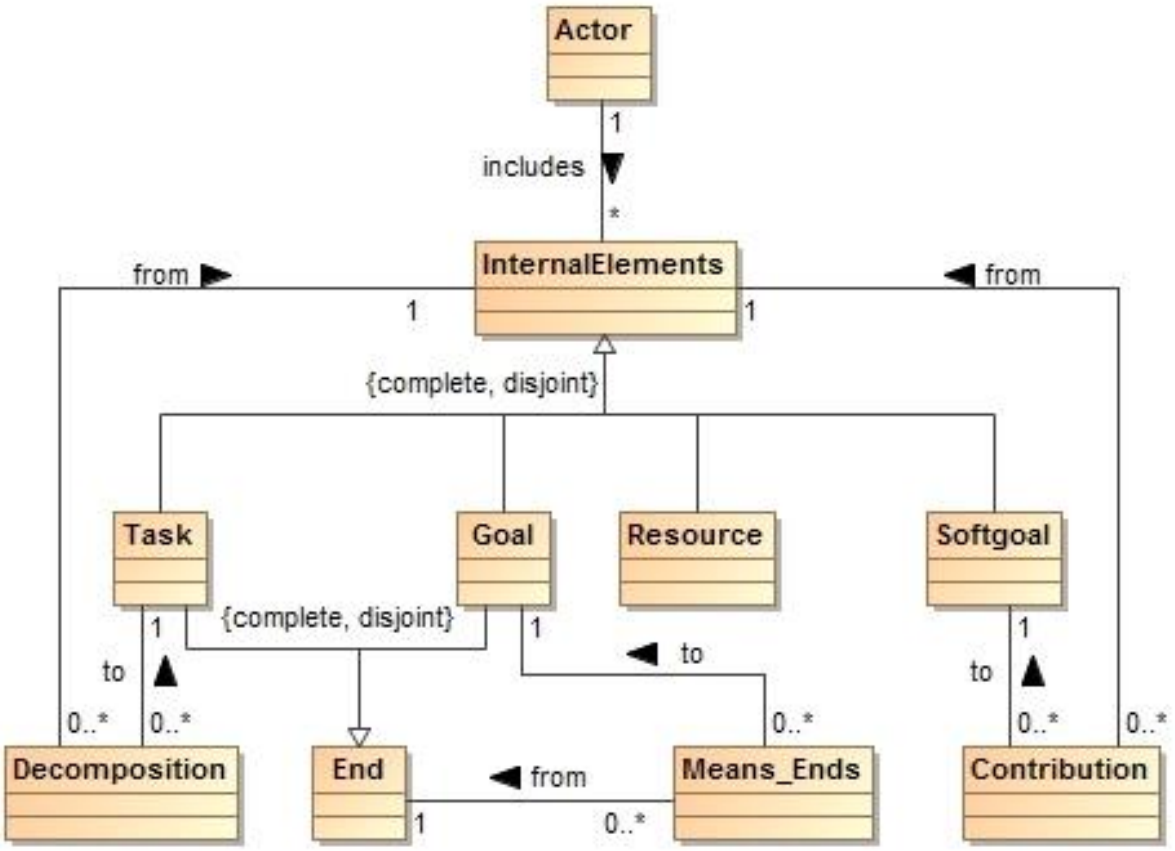

Figure 4. - Syntax of SR model, adapted from (Ayala *et al*., 2005).

The *decomposition* links can be *open*, *committed* or *critical*. Similar to the SD model, *committed* means the routine will most likely fail if the element fails, *open* means that the *routine* could fail but not necessarily, and *critical* means that there is no other way to succeed.

A *mean-ends* link represents the link between an *end* and the means of obtaining it. The *end* could be a *goal* to achieve, a *task* to be performed, a *resource* to be produced or a *softgoal* to be satisfied. The means is usually represented as a *task*. The *means-ends* links are also categorized.

In the SR model, *routine* is defined as a set of choices. A *routine* is a single link from *means* node to the *ends* node, meaning it only represents one particular course of action among the multiple choices.

There is also a *rule* concept in the SR model. A rule is a *means-ends* link that is not yet bound. The *rule* consists of a condition, a mean, and an *end*. The *means-end* link is an application of the *rule*, if it is decided that it follows the condition defined in the *rule*.

The rational elements as the condition in the *rules* are supplementary information and is not directly part of the *means-ends* hierarchy. They just provide support and are modelled as beliefs.

3.4 The *i** model examples

The example is made to cover most of the main concepts of the *i** and is not made for any specific company and is all made up. The *i** model is made up by SD (Figure 5) and SR (Figure 6) models. The words in *italic* are concepts, and words or phrases in arial font are taken directly from the example to give a better explanation.

Strategic Dependency model's example. The first task to do is to define the main *actor*. In this example (Figure 5) of a retail chain and its online shopping system, the main actor is the management of the retail chain. In the model it is in the middle (*actor* Retail chains management).

After the main *actor* is defined, it is needed to add some more *actors* to depend on the retail chains management or *actors* to be dependent on by the management. For this, there is defined five more actors (*actors* Customers, Software engineers, Testers, Delivery and warehouse workers, Construction company).

When all the *actors* needed are defined, adding the *dependencies* can start. The motivation to go on and do the changes comes from the customers (*actor* Customers). The customers depend on the management (*actor* Retail chains management) with the online shopping system that is not existing at the moment of making this model (*goal dependency* Service to buy goods). In SD model, it is not defined how the dependum is achieved, it just is defined to be important for the customers. Also the management depends on the customers to use the built system after it is released for public use, so there is a *goal dependency* from the management to customers (*goal dependency* Online system used).

The next *actors* to be described is software engineers (*actor Software* engineers) and also the testers (*actor* Testers) as they will work together on the system building. Both software engineers and testers depend on the management (*actor* Retail chains management) with the system requirements (*resource dependency* System requirements) for the system that is needed to be built. The engineers also depend on the management for the funds to build the system *(resource dependency* Funds). The dependum system requirements is defined two times in the model, because one dependum can only have one *dependee* and *depender*. The management depends on the engineers to build the system (*task dependency* Build online system) and the final product (*resource dependency* Functional online shopping system). While the engineers are building the system, they depend on the testers to test the system they are making (*goal dependency* System tested) while the testers have a dependency on the software engineers to provide them with the system to test (*resource dependency* Online shopping system).

The fourth *actors* in this model are delivery and warehouse workers (*actor* Delivery and Warehouse workers). This *actor* only depends on the management (*actor* Retail chains management) to hire them and provide a contract on the delivery service (*resource dependency* Delivery service agreement). To get that, the retail chain has to request the delivery company for its service (*task dependency* Request for delivery service).

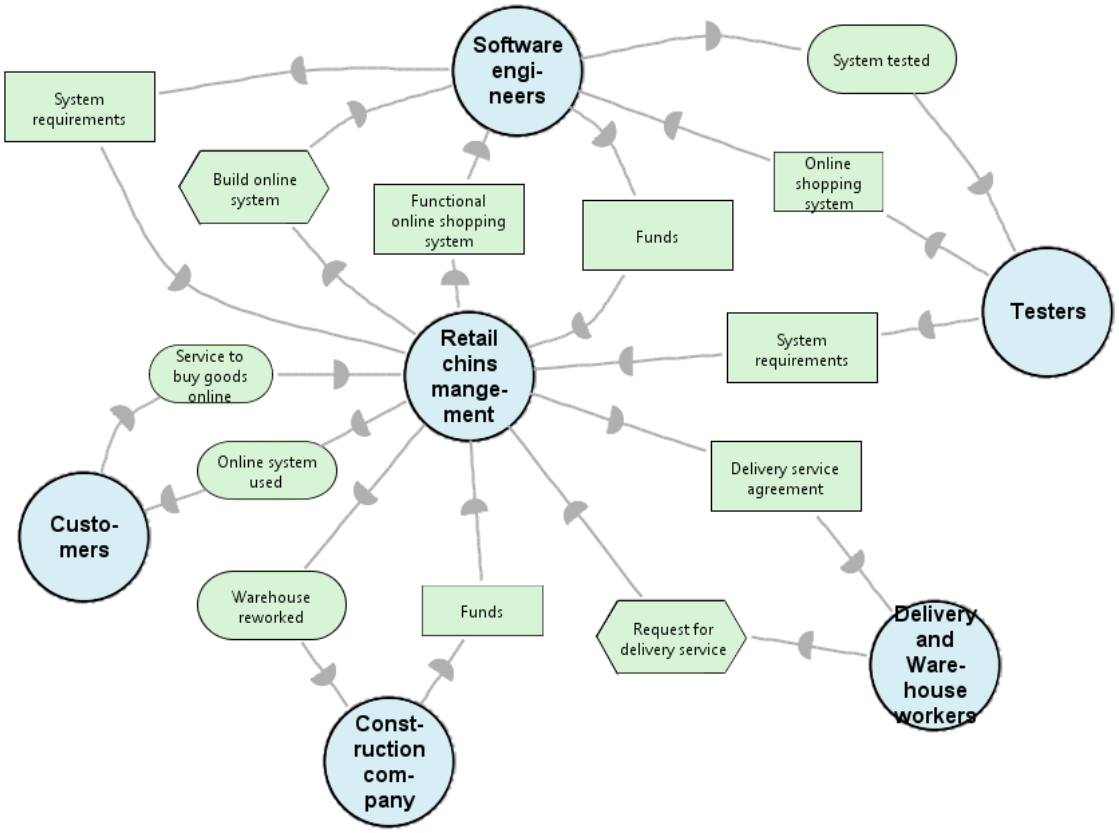

Figure 5. - Example of Strategic Dependency model.

The last *actor* in the SD model is the construction company (*actor* Construction company). The company has to rework the warehouses of the retail chain to be suitable for delivery service. The construction company depends on the management for the funds (*resource dependency* Funds) to make the rework. The management depends on the company the actually finish the rework (*goal dependency* Warehouses reworked).

Strategic Rationale model's example. The example of SR model (Figure 6) has the same *actors* defined as the SD model (Figure 5) had (*actors* Retail chains management, Customers, Software engineers, Testers, Delivery and warehouse workers, Construction company). The difference in those *actors* are their *boundaries* - grey circles with the *actor's* name on top of them. Inside those *boundaries* are defined the internal elements of the actor. The elements show what the *actor's goals* are and which *task* they need to perform.

Starting out with the customers (*actor* Customers). Their only *goal* is to buy their goods online and then getting these products delivered to them (*goal* Buy groceries online and have them delivered). To achieve this *goal*, the customers need to address the retail chain about their *goal* (*resource dependency* Customers' wishes).

After the customers have send out the wish to have an online shopping system, the management (*actor* Retail chains management) will get them (*task* Get the customers' wishes). With a *means-ends* link, one of the managements *goals* (*goal* Build the online shopping system with delivery) is started. The *goal* will be achieved by completing three of the tasks attached

to it (*tasks* Hire software engineers and testers, Hire a construction company, Hire warehouse and delivery workers).

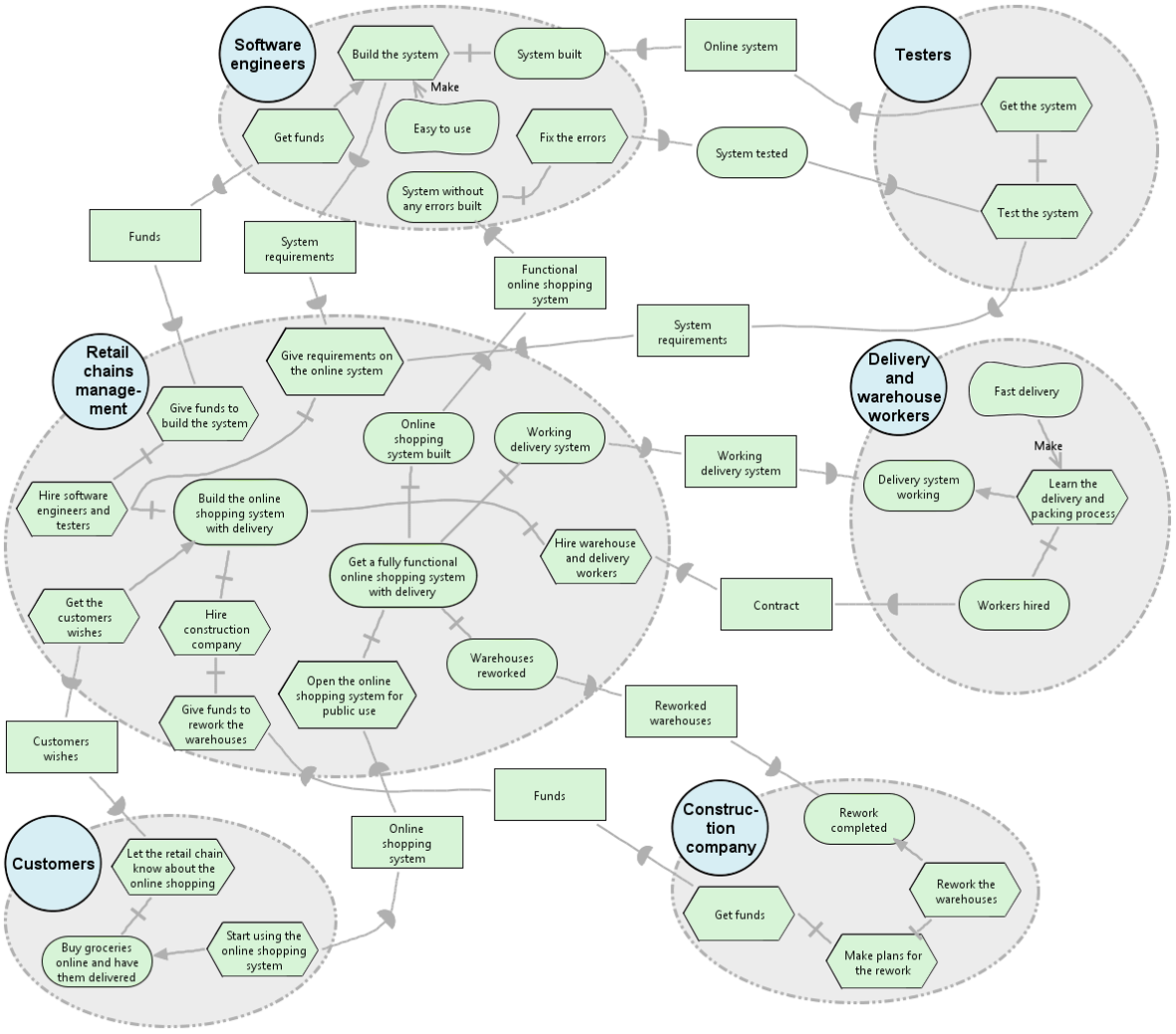

Figure 6. - Example of Strategic Rationale model.

The first *task* is to hire the engineers and testers (*task* Hire software engineers and testers). After the managements has hired them, they need to give the engineers (*actor* Software engineers) the funds to build the system (task Give funds to build the system) and also the requirements for the new system (*task* Give requirements on the online system), which both the engineers and testers are depending on.

The software engineers (*actor* Software engineers) need the funds and the requirements to build the system (*goal* System built). After they have the funds and requirements, they can start building the system (*task* build the system), which is required to achieve the *goal*. The *task* has a *softgoal* linked to it, which states that the system needs to be easy to use (*softgoal* Easy to use). *Softgoal* in this case is used to describe the tasks outcome and is not something very specifically defined. When the system is built, the software engineers will give the online system (*resource* dependency Online system) to the testers for testing purposes. Testers

main *task* is to test the system, which the engineers have made (*task* Test the system). After which the engineers will need to fix the errors found while testing (*task* Fix the errors). Completing this *task* will achieve the software engineers *goal* to have the system built (*goal* System without any errors built).

The second management's *task* was to hire a construction company (*task* Hire a construction company). After the company (*actor* Construction company) is hired, the management will need to fund the rework on the warehouses (*task* Give funds to rework the warehouses). Having the funds (*task* Get funds) the construction company can start planning the rework (*task* Make plans for the rework). When the plans are made, the rework will be made according to those plans (*task* Rework the warehouses). Completing this *task* too, the construction company's *goal* (*goal* Rework complete) is achieved and the warehouses can be handed over to the retail chain.

The third *task* for the management was to hire new workers and a delivery company (*task* Hire warehouse and delivery workers). This is done by making a contract (*resource* dependency Contract) with the construction company and if needed, also hiring some more workers to the warehouse. When the contract is made, the delivery company's and the new warehouse workers can start learning the process of the delivery (*task* Learn the delivery and packing process). The delivery process needs to be fast, so there is a softgoal linked to that task (*softgoal* Fast delivery). When the process is learned, the delivery system is working and ready for use (*goal* Delivery system working).

After all the *actors* (*actors* Software engineers, Testers, Delivery and warehouse workers, Construction company) have achieved their own *goals*, the management will need to accept their works (*goals* Online shopping system built, Working delivery system, warehouses reworked). Fulfilling those goals will let the management achieve one of his goals to have the online shopping system with delivery (*goal* Get a fully functional online shopping system with delivery). After achieving this *goal*, the management has one final *task*, which is to open the new shopping system to their customers (*task* Open the online shopping system for public use).

Having opened the online shopping for public used, the customers' *goal* of buying their goods online and then getting them delivered is achieved (*goal* Buy groceries online and have them delivered).

3.5 Summary

In this chapter the main concepts of *i** are described while giving an overview on the semantics and abstract syntax. For better understanding and also for later evaluation, an example of *i** is made and also described. Next chapter will provide guidelines to make BPMN from the BMM and *i** models and also the BPMN deriving from the modelling examples is presented.

4 BPMN deriving from the BMM and *i** models

With the BMM and *i** models comes Business Process Model and Notation (BPMN) (Figure 7). The words in *italic* are concepts, and words or phrases in arial font are taken directly from the example to give a better explanation. The table containing all the BPMN concepts and the corresponding BMM and *i** concepts can be found in the appendix (Appendix 1).

4.1 Getting BPMN from BMM

To get the BPMN model from BMM model can be done by following the steps below:

1. Put all the roles mentioned in BMM as lanes to the BPMN
2. Add mission to the BPMN as start event and vision as end event
3. Add the tactics as tasks to the BPMN, make multiple smaller tasks if needed
4. Add goals as intermediate events to the BPMN, make multiple smaller tasks if needed

These are the main points to follow, it may be necessary to do some extra work to fit all the objects from BMM to BPMN.

4.2 Getting BPMN from *i**

To get the BPMN model from *i** models can be done by following the steps below:

1. Put all the actors as lanes in the BPMN
2. Find the cause goal and the ending goal from the *i** model
3. Put the found goals as start and end events in the BPMN
4. Follow the *i** model and add all the tasks to the BPMN and the other goals as intermediate events
5. Dependences between actors are used as flows between lines in the BPMN

These are the main points to follow, it may be necessary to do some extra work to fit all the objects from *i** to BPMN.

4.3 Derived BPMN model

After both, BMM and *i** models are complete, it is possible to make a BPMN deriving from those two. The starting point in the BPMN will be the customers wish to start buying groceries online and having them delivered from the SR model (*goal* Buy groceries online and have them delivered), same can be found on the BMM, where it is defined as an assessment (*assessment* Customers would like an online shopping system with fast delivery). The first *task* by customers is to ask the retail chain to start an online shopping system (*task* Ask retail chain to launch an online shopping system), which starts the managements (lane management) work.

After the customers' wishes are gotten by the retail chain and the management has started working on building the online shopping system, there will be three parallel *tasks* to do. Those *tasks* are to give funds to rework the warehouses, hire a delivery company and warehouse workers, and give funds and requirements to build the online system. All of the *tasks* are deriving from the SR model, where they are also defined as the retail chain's management's tasks, similar *tactics* can be seen in the BMM.

After the construction company is hired, the first *task* will let the construction company (*lane* Construction company) to start planning their work on the warehouses (*task* Plan the warehouse rework). After the planning is done, the company can start on the rework (*task* Do the warehouse rework), which when done, will be the basis to achieve the construction company's *goal* of completing the rework (*intermediate event* warehouse rework completed). Like before, the *tasks* are a directly taken from the SR model, where they are defined in (*actor* Delivery and warehouse workers). The last *task* is to hand over the reworked warehouses to the retail chain (*task* Hand over the warehouses).

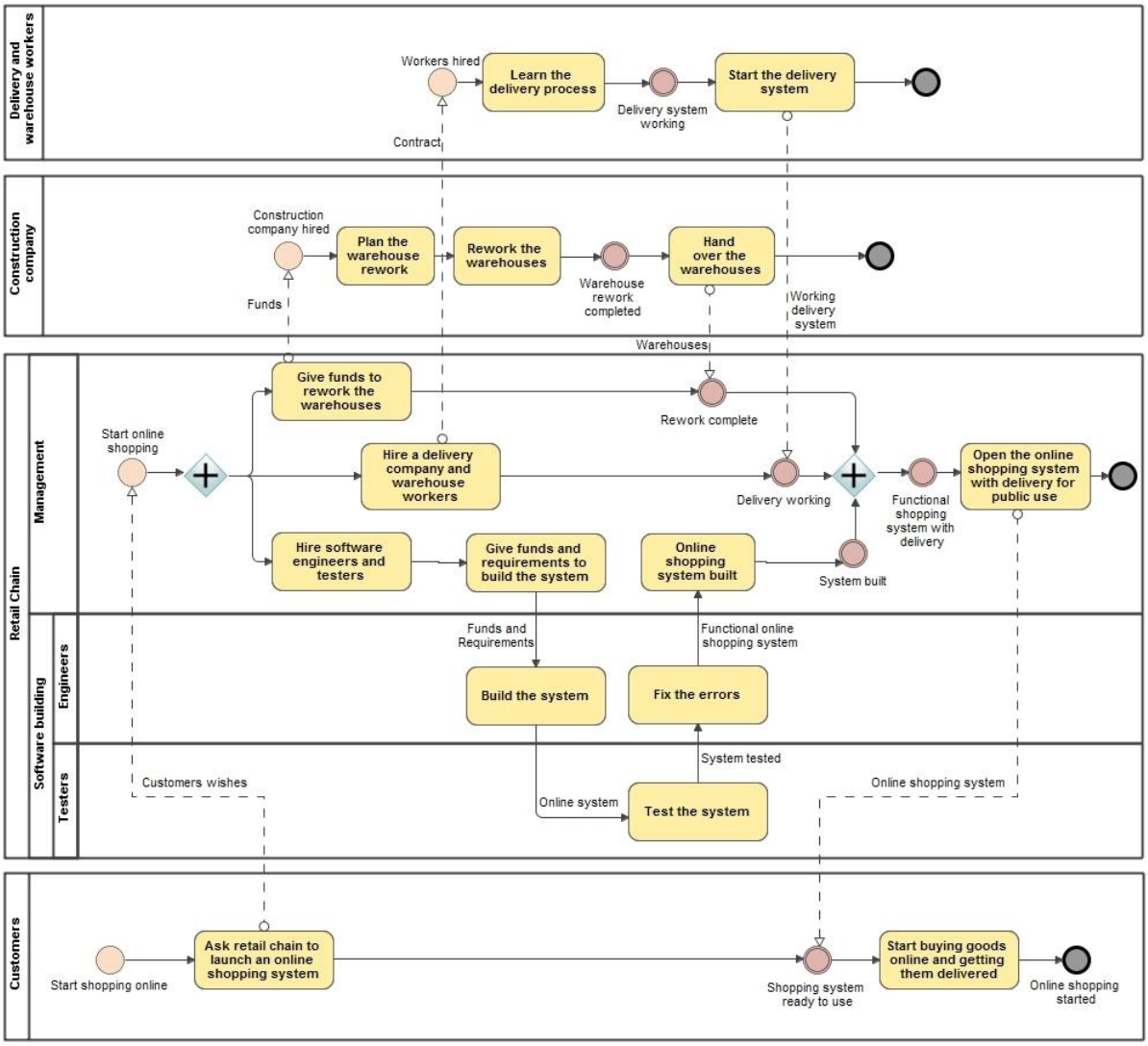

Figure 7. - BPMN based on BMM and *i** models.

The second of the management's *task* is to hire new workers and a delivery company (*task* Hire delivery and warehouse workers). When the workers are hired (*start event* Workers hired), they will start learning the process of the delivery and with that the process of packing the goods bought by the customers (*task* Learn the delivery process). If the *task* is complete, the delivery system can be started (*task* Start the delivery system) and will be ready to use (*intermediate event* Delivery system working). The *task* of learning the delivery process is

defined as a *task* in *i** and as *tactic* in BMM (*task* Learn the delivery and packing process; *tactic* Get the workers to be more efficient) and the intermediate goal defined as a goal in the SR model (*goal* Delivery system working).

The last of management's parallel *tasks* is to start building the online shopping system by hiring the software engineers and testers (*task* Hire software engineers and testers) and giving them funds and the requirements (*task* Give funds and requirements to build the online system) to the software engineers (*lane* Engineers). When the system is built (*task* Build the system), the testers (*lane* testers) will need to test it (*task* Test the system) and give the testing information to engineers to repair the errors found (*task* Fix the errors). After the errors are fixed (*task* Repair the errors), the built system can be handed over to the management (*task* Online shopping system built) and the intermediate event is achieved (*intermediate event* System built). All the *tasks* in those lanes are also defined in the SR model and in BMM, from where they are taken from and put into the BPMN model.

Having achieved all the *intermediate events* (*intermediate event* Delivery working, Rework complete, System started) the management's parallel gateway is passed. After this only thing to do for the management is to open the system for public use (*task* Open the online shopping system with delivery for public use), which is also a *task* in the SR model (*task* Open the online shopping system with delivery for public use).

Last *task* to do is in the customers' lane. The customer will need to start using the new online shopping system provided by the retail chain. After the customers are able to and are using the system, the end is achieved (*end event* Online shopping started). The *end event* is defined as a *goal* (*goal* Buy groceries online and have them delivered) in the SR model (*actor* customers).

4.4 Summary

In this chapter the process of getting BPMN from BMM and *i** is described. After that, the BPMN deriving from BMM and *i** is presented and described. Next chapter will give an overview on the Language and Model Quality Evaluation Framework and will define the criteria for later evaluation.

5 Language and Model Quality Evaluation Framework

This chapter gives an overview of the language and model evaluation framework. In addition to the framework overview, the criteria for the comparison is defined. The framework overview is based on (Krogstie, 2012). The criteria is deriving from (Krogstie, 2012; Matulevičius and Heymans, 2007).

5.1 Language Quality Evaluation Framework

The languages compared are BMM (Chapter 2) and *i** (Chapter 3). The language used for this comparison is semiotic quality framework (SEQUAL) for language quality.

SEQUAL for language quality. According to (Krogstie, 2012) there are a total of six areas (Figure 8) that can be used to compare two modelling languages. *Domain appropriateness* evaluates the basics of the modelling language and how useful they are to use. It is mostly used to compare the different concepts in the modelling language. *Comprehensibility appropriateness* is used to understand the social interpretations. Mostly relates to the people working on the model. *Participant appropriateness* is used to evaluate the participants' knowledge on the language. Participants in this case are the people working with the model and using it after it is completed. This makes the participants appropriateness important for companies as the end user of the model is not someone who made it, but some other person in the company. *Modeller appropriateness* evaluates the knowledge of the modeller with the modelling languages. Can be used to find out how easy the language is to use or how much effort is needed to understand it. *Tool appropriateness* is used to evaluate the languages tools. Mainly used on the tools supporting the languages used and gives the verdict on its executability and analysability. Lastly, *organisational appropriateness* shows the relations between the language and the organization using it for work.

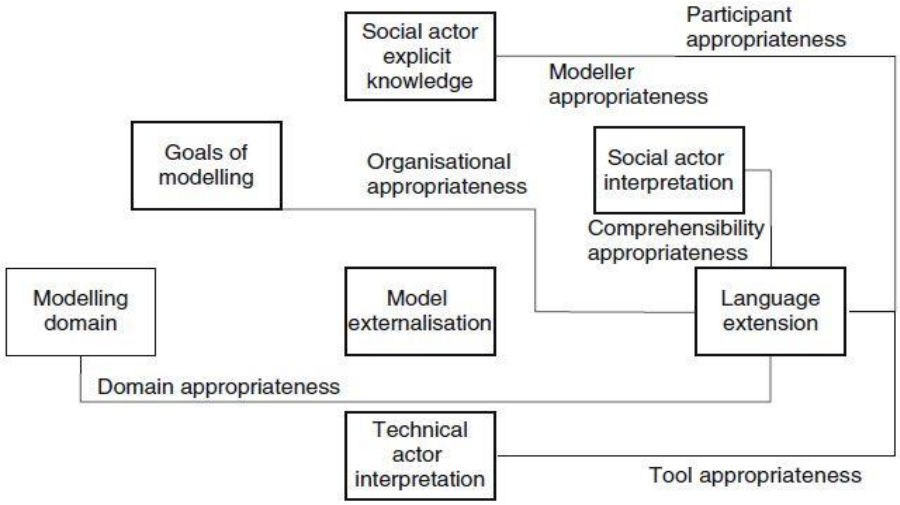

Figure 8. - SEQUAL for language quality (Krogstie, 2012).

Criteria for language quality evaluation. The criteria is deriving from (Krogstie, 2012; Matulevičius and Heymans, 2007).

C.1. Number of constructs shows the number of different constructs defined in the language.

C.2. Criteria similar constructs gives an overview of how much one language has similar constructs with the other language.

C.3. Graphical representation shows how good the language graphical side is. To follow this criteria, the language needs to have a graphical representation for each construct it has. This also depends on the tools used to make the model.

C.4. Well-defined constructs show how good the graphical representations are. All the constructs should be easily distinguishable and be easy to understand.

C.5. Expressiveness power shows the relationship between views and the number of constructs.

C.6. Criteria number of views covered is an overview of the modelling perspectives on the corresponding languages. The description on the perspectives is based on (Krogstie, 2012). The perspectives are behavioural, functional, structural, goal and rule, object, communication, and actor and role. The discussion for view coverage is in Section 6.1. The perspectives, which have no similarities with BMM nor *i** are not described in the conclusive table (Table 4).

C.7. Automated analysis method will show if the tool used for the modelling supports automated analysis on the model.

C.8. Programmable infrastructure will show if the tool supports generating code for programming by taking the model as template.

C.9. Formal semantics will show if the model could be misunderstood from a mathematical point of view.

C.10. Available methodologies will show if there is published methodologies for specific purposes.

C.11. Tool support will show if the tool has support for usage of any kind.

C.12. Community support will show if the language has a community of users.

C.13. Free use will show if the language is free to use.

5.2 Model Quality Evaluation Framework

Comparing two already made models is done by using a framework for evaluating the quality of models based on SEQUAL for model quality. There are seven main concepts in SEQUAL (Figure 9) and they are described by Krogstie (2012).

SEQUAL for model quality. The first concept is *physical quality*. It is mostly affected by the tool used for making the model. The example model of BMM is made by using Visual Paradigm and *i** models are made with OpenOME. *Empirical quality* is the usage of visuals. It is used to give a rating to colour usage, font size and other visual qualities of the models. *Semantic* and *syntactic quality* is used to evaluating the correctness and completeness of the model. The fourth

quality is *pragmatic*. It is mostly used on the audience and users of the model. For the examples in this paper, they are not meant as a real work for a company and because of that, the pragmatic quality has no aspects defined. The last quality used is *deontic*. This mostly focuses on the financial aspects of the model and how much funds it needed, also the goal and its qualities. Again, the models are just examples and do not have financial aspect in reality.

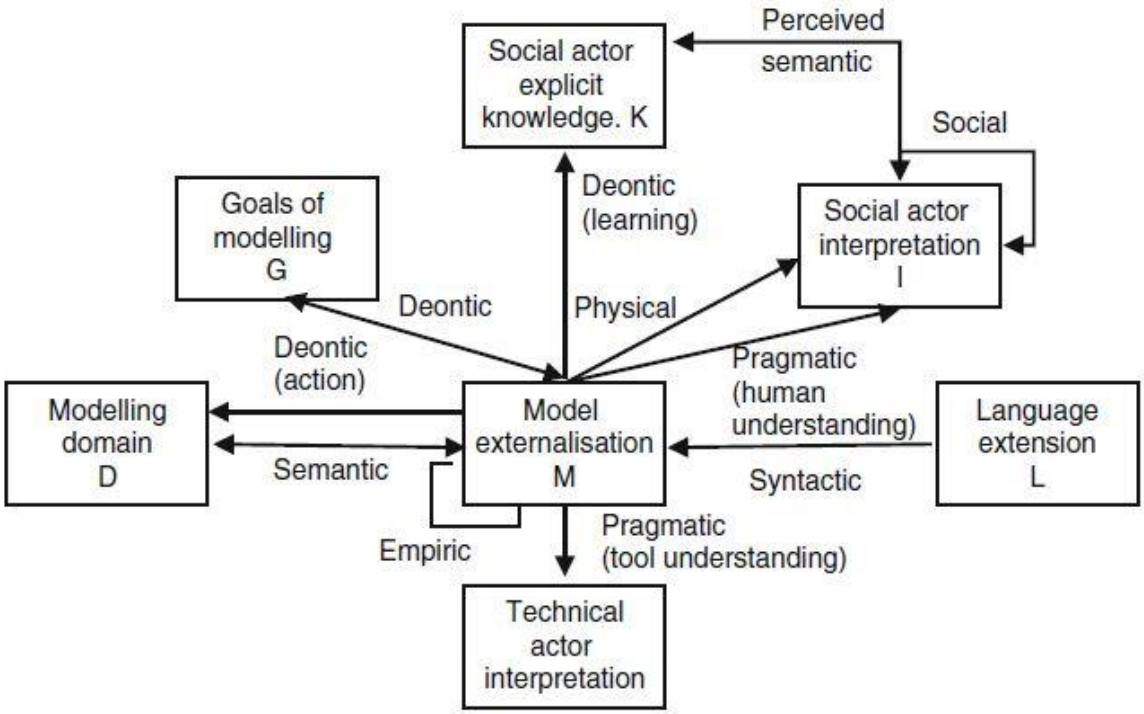

Figure 9. - SEQUAL for model quality (Krogstie, 2012).

SEQUAL can be used for a very specific evaluation. In this case, the evaluation will be only on the main aspects of the models. The overview of the properties to be evaluated are described next that is based on (Krogstie, 2012).

Criteria for model quality evaluation. The criteria is deriving from (Krogstie, 2012; Matulevičius and Heymans, 2007).

Physical quality evaluates the tools and the storage of the files. The tools should let the user freely copy over existing objects and reuse them.

Next is empirical quality for the visuals of the model. The first property defined is colours used. According to (Krogstie, 2012), the normal usage of colours is at four different, with the maximum to be used at seven. With the different colours, the consistency of them needs to be right. Same colours should not be used for different objects and same objects should not have different colours. It makes the model easier to read and understand. The limitation on the colours is made for people who have some kind of colour-blindness, where smaller amount makes it easier to difference between them. Font size and size of different objects is also empirical. The font size should be easy to read for most people. The size of objects makes some of them look out more, bigger objects are seen first and thus seem more important. To avoid

confusion, the objects should be similar in size and the bigger elements actually be more important. Also the objects in the middle are mostly noticed first.

Syntactic quality is the correspondence between the language used in the model and the language defined in the concepts of the modelling language. Two kinds of errors can be made in the syntactic aspect (Krogstie, 2012). First kind of error is the syntactic invalidly, where constructs or words are used in the model that the language itself does not have. The second is syntactic incompleteness. In this error, the model does not have constructs, which are defined to be required by the language used.

Semantic quality is the correspondence between the model and the modelling domain. While a lot of properties can be defined for semantic quality, there are two main ones: validity and completeness. To be valid, the statements in the model must be defined correctly and be relevant. To be complete, the model has to contain all the statements that are correct and relevant for the problem. To be valid and complete, the model has to be consistent. Consistence requires the objects not to conflict with each other.

Deontic quality measures the goal and its qualities. In this example, deontic quality is used to see, how understandable the goal of this model is. To get the perfect score, the goal needs to be easy to understand and also easy to see in the model.

5.3 Summary

In this chapter an overview on the Language and Model Quality Evaluation Framework (SEQUAL) is given and the criteria for later evaluation is defined. Next chapter will provide the evaluations on language and model quality, based on the criteria defined in this chapter. Also the observations of using BMM and *i** to make BPMN are discussed.

6 Comparison

This chapter is about the comparison between BMM and *i** in both, language and modelling aspect. The criteria and methods for the comparison are based on (Krogstie 2012) and are defined in Chapter 5.

6.1 Language evaluation

The results of evaluation on BMM and *i** languages can be seen in Table 1. The discussion on the results is after the table.

Table 1. Criteria for the BMM and *i** language.

	Requirement		Appr.	BMM	*i**
C.1	Number of constructs		D, C	15	13
C.2	Similar constructs		D	27%	31%
C.3	Graphical representation		C	79%	65%
C.4	Well-defined constructs		D, C	Partially	Partially
C.5	Expressiveness power	Behavioural	D, C	3	6
		Goal and Rule	D, C	9	6
		Actor and role	D, C	0	12
C.6	Number of views covered		D	1.5	2.5
C.7	Automated analysis methods		T	No	No
C.8	Programmable infrastructures		T	No	No
C.9	Formal semantics		T	No	No
C.10	Available methodologies		O	Yes	Yes
C.11	Tool support		O	Yes	Yes
C.12	Community support		O	Yes	Yes
C.13	Free use		O	Yes	Yes

Number of constructs

Number of constructs shows the amount of different concepts in the language. BMM has a total of 15 concepts defined (Figure 10) and *i** has 13 (Figure 11).

Similar constructs

Although both languages are used to give rationale for business, they do not have many similar concepts defined. The similar contracts between BMM and *i** are described in Table 2. For example, both the models have *goal* class. For BMM it is a smaller part of something bigger (*vision*), but in *i** the *goal* is more similar to *vision* in the documentation.

Table 2. Similar constructs between BMM and *i**.

BMM construct		***i** constructs**	
Construct	**Description**	**Construct**	**Description**
Vision	Overall image of the desired end	Goal	A specific condition that the actor wants to achieve
Goal and Objective	Goal is more narrow version of vision to make it easier to follow, objective shows the steps takes towards goal	Softgoal	More narrow versions of goal to make it easier to understand.
Course of action	Actions taken towards achieving the end	Task	Specific way of doing something to achieve the goal
Influencer	Somebody or something to cause the changes	Actor	Somebody, who want to achieve the goals

There are also constructs with same or very similar names defined for both, BMM and *i**. Those can be seen in Table 3.

Using the information given in Tables 2 and 3, it is possible to get the overall semantic similarity for the languages compared. BMM has similarities with *i** in 27% of its' total constructs and *i** 31% with BMM.

Table 3. Similar name constructs with different meaning between BMM and *i**.

Construct	**BMM**	***i****
End	Something that the enterprise wants to be	Something to be achieved (goal), performed (task) or produced (resource)
Means	Steps needed to get to the end	Means-Ends – Link between two nodes to show how end node will be achieved
Goal	A narrow version of the vision to make it easier to follow	A specific condition that the actor wants to achieve

Graphical representation.

BMM is made with Visual Paradigm and *i** with OpenOME. The BMM model has a separate representation for most of its constructs (approx. 79%) (Figure 10). Abstract concepts (*i.e. directive, desired results etc.*) do not have a graphical representation. The *i** language has defined a way to represent most the links and nodes (approx. 65%) (Figure 11). Like for BMM, there are a few of abstract constructs (*i.e. internal elements, dependum etc.*) that do not have a

graphical representation. The visuals can be seen in the examples (Figure 2, Figure 5, and Figure 6).

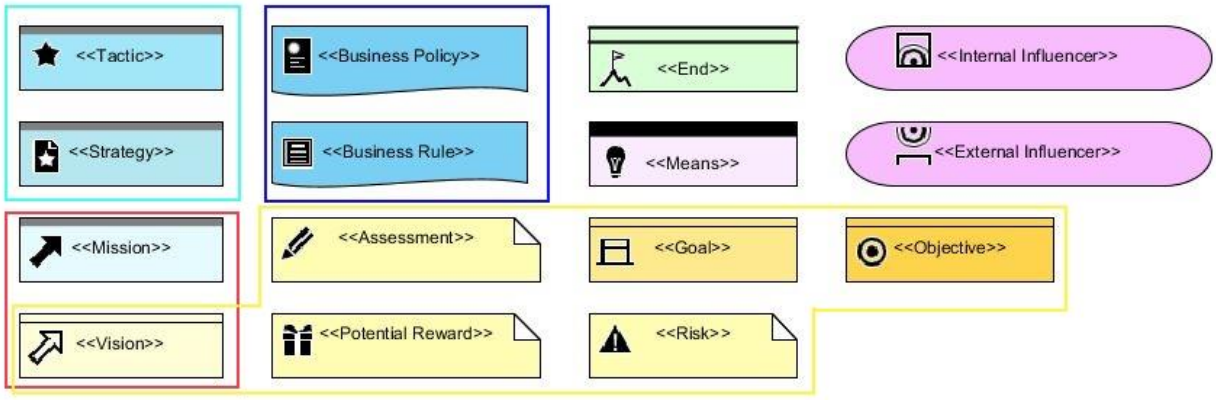

Figure 10. - BMM's graphical constructs (Visual Paradigm).

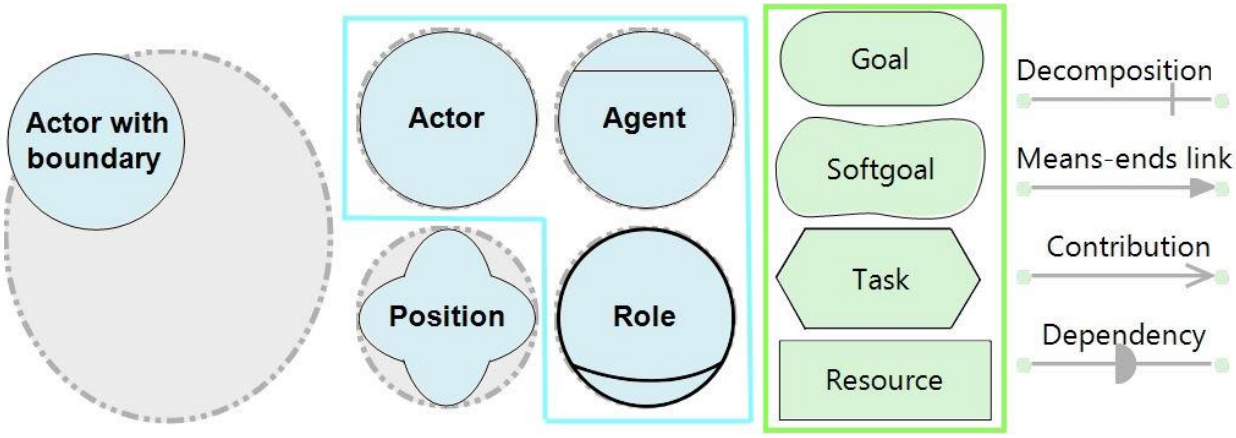

Figure 11. - *i** graphical constructs (OpenOME).

Well-defined constructs

BMM has multiple very similar representation to different concepts. There are two different blue box nodes (tactic and strategy), which have very slight difference in colour and differ from each other by the small icon that also are similar because of the star shape. The business rule and policy (light blue outline on Figure 11) have almost the same representation, only differ from each other by the slight difference in the icon. The same can be said for the risk, reward, and assessment, and goal and objective (yellow outline on Figure 10). Mission and vision (red outline on Figure 10) have the same icon, an arrow, which is filled in on mission. Influencer, on the other hand, has two different representations depending if it is internal or external. This means that the BMM has poorly defined constructs, almost all, eight out of twelve used concepts, of them have big similarities with another construct. In the *i** language the actors and internal elements have different colours and shapes from each other. A not specified actor is blue circle, when specified, it may be a blue circle with a line through it or a cross shape. Internal elements have the same node type as the dependencies, making it hard to understand. On the other hand, the same node will represent the same concept. For example, the task node in an actor's boundary is the same as it is on the task dependency link, differing from each other by just the name written on them. All the dependencies and internal elements differ from each

other by the shape of the node. The colours are the same for them, but the shape still makes it easy to understand. The colours are the addition by the tool used, *i** was originally made to depend on shapes and not colours. The similar constructs can be seen on Figure 11, where the similar objects have same colour lining around them.

Expressiveness power

BMM construct and view relationship can be seen on Figure 12 and *i** relationships are on Figure 13. The views for both languages are described in Section 5.

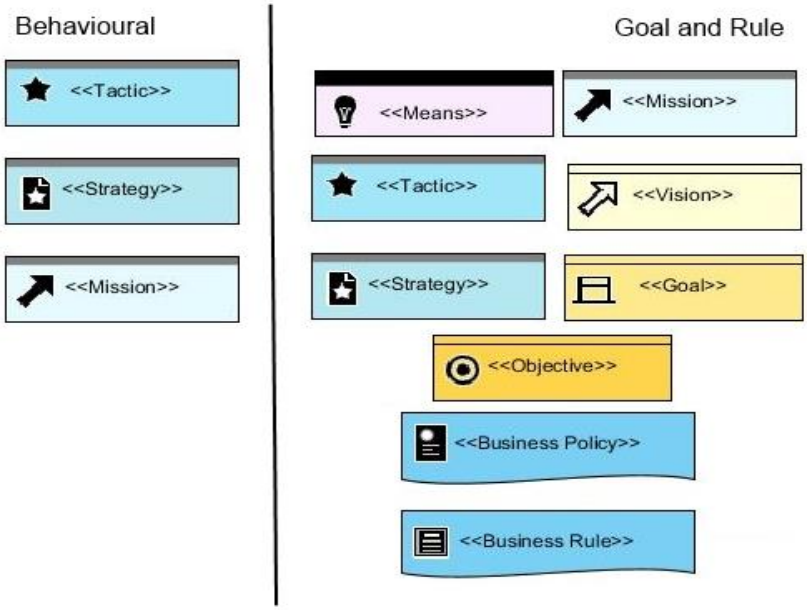

Figure 12. - Construct and view relationship of BMM.

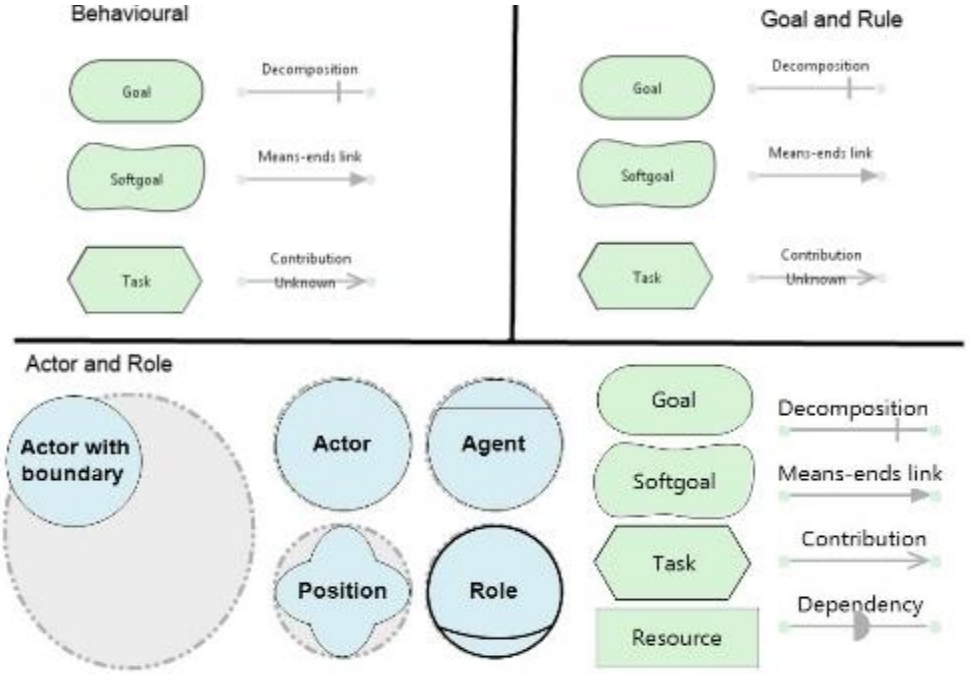

Figure 13. - Construct and view relationship of *i**.

View coverage

Behavioural perspective is mostly used in languages which consist of states and the movements between the states, where events trigger the movements. An example of this is finite state machine. BMM is partially behavioural. The movement between tactics, strategies, and vision can be seen as a movement between states. To go from a tactic to strategy, an event is needed and from strategy to mission too. In this case the event is to implement the strategy. The *i** language is similar to BMM in this aspect, it is partially behavioural. The goals are states and the tasks show how they are achieved.

Functional perspective shows the flow in systems, where an input is transformed into something else and then given out as an output. Neither BMM nor *i** work this way.

Structural perspective describes the structure of a system. It consists of entities and relationships that can be classified. Mostly used in object-orientated approaches Neither BMM nor *i** work this way.

In goal and rule perspective languages, the goal is achieved by following the rules described. The goals have hierarchy, meaning the main goal can have smaller goals to help it being achieved. In BMM the main goal is the vision and it is amplified by goals. It is also possible to write rules and policies into the model, making it goal and rule perspective. The same can be said for *i**, where goals can have links between them and are hierarchal. Also, tasks are used to describe how the goals will be achieved.

Object perspective is the basis for object-orientated modelling languages. The main aspects of it are the objects, which is in a class that can have multiple objects in it, and the processes between the objects. Neither BMM nor *i** work this way.

Table 4. Perspectives for the BMM and *i** language.

Perspective	**BMM**		***i****	
Behavioural	Partial	Movement between tactics and strategies (events) and mission (goal).	Partial	Movement between tasks (event) and goals (goal).
Goal and Rule	Yes	Vision is divided into goals; business rules and policies to follow. Means will show the way of achieving the goals.	Yes	Goals can be divided into smaller goals or softgoals; tasks define the way to achieve the goal.
Actor and Role	No	Does not have an actor nor role concept.	Yes	Main concept is actor who has dependencies on other actors

Communication perspective is used in languages, where the flow of the model is based on communication. Has the roles of a speaker and hearer, between whom the communication takes place. Neither BMM nor *i** work this way.

In actor and role perspective languages, the main aspect is the actor or the role. The model is worked around the actors or roles, showing their dependencies. The *i** language was made to follow this ruleset and is one of the most used languages in actor and role perspective. BMM, on the other hand does not follow this.

6.2 BMM and *i** models quality

This Section gives an overview of BMM and *i** model evaluation. The criteria used is defined in Chapter 5.

BMM model's quality evaluation

The model files are saved electronically in Visual Paradigm format. One physical qualities property is also the reusability of the model. When a new model is in making, the old one can be opened too and the parts can be copied over, making it easy to reuse some of the old model's parts.

The colours used in the BMM is 11, making it way over the maximum recommended by Krogstie (2012). Also, many of the colours used are very similar to each other. For example, there are four different yellow coloured boxed (Figure 10) that may look very similar at first look. Size 12 font is used in this example. The first elements seen on the model are means and end. They are way bigger in size than other objects and even contain smaller objects inside them. This can be both, good and bad. Good in the aspect that the goal of the model is in the end object and the way to achieve it is defined in the means. But the cause of the modelling is defined in the influencer nodes that are smaller and will not be seen at first. The cause or motivation to do something is the main aspect of the model and with such visuals, it may not look like it.

All the words seen on the BMM example are part of the BMM language, making it valid. This excludes the descriptions on the nodes, the blank objects can be seen on Figure 10. The blank nodes with the words showing their type are part of the Visual Paradigm and because of this, they cannot be changed. Also all the links between the nodes have some words written on them, which again are there by default, making them to be part of the language. The model is also complete in syntactic aspect, it has all the four main concepts in it: influencers, assessments, means, and end.

The model is valid in semantic aspect. All the statements are correct and carry over some information about the problem, making them relevant. The model, on the other hand, is not complete. For example, it is possible to define more business rules and policies about the working conditions or workers. There is no problems with consistence, all the statements work together and do not state the opposites.

The goal of the model is easily seen, it is on the end element that is one of the biggest in the model. It is short in description and makes it easy to understand.

*i** models' quality evaluation

The model is stored electronically in the format for OpenOME. Like BMM, the object from one model can be easily copied over the second one, making it easy to modify and update.

Colours used in the *i** models is four, excluding the font colours. This is the same number that is recommended by Krogstie (2012). The font size is 12 in the electronical copy of the model. The size of the different actors differs a lot. This is caused by the number of internal elements in them. The biggest one of them is the retail chains management, which is also in the middle of the model. This can be considered good, as the management is the most important actor, that controls the flow of the work and has dependencies on all other actors and all the other actors depend on the management. The sizes for other actors and elements are quite similar.

All the words used in the *i** models are the description of the elements and are defined by the modeller. Only words that are there by default are the contribution links, with the word describing the type of the contribution link. Those words are part of the language. The model is complete in syntactically as the actors have dependencies on each other and also have goals to be achieved and the way of achieving them as tasks.

The statements on the *i** models are there to carry over some information about the goal, task or actor. This makes the model valid semantically. The models cannot be considered complete. It is possible to define more softgoals to specify some aspects, or even new tasks to make the achieving of the goal easier. On the other hand, it is consistent, as the objects do not have conflicts with other objects in the model.

The main goal of the model is hard to see on the first glance. There are multiple goals defined and finding the main one is problematic without any previous information.

6.3 BPMN model deriving from BMM and *i**

Evaluation for getting BPMN from BMM and *i** is done by using the mapping table from the BMM and *i** models to the BPMN model. The table for this comparison (Appendix 1) has all the objects seen on BPMN model and their corresponding objects from BMM and *i** models, where the mapping of concepts from one language to another is seen. Table 5 gives the results from appendix 1 and shows the mapping of concepts in an abstract way with their description.

For BMM and BPMN the best similarities are with means (tactic and strategy) to task and ends (goal and vision) to events. For example, *Tactic* Rework the warehouses for better delivery times is a direct mapping to BPMN's *Task* Rework the warehouses. Similar example can be found for the ends to event mapping, for example, *Vision* Online shopping and delivery working to *End event* Online shopping started. More examples can be found in the mapping table (Appendix 1). Also, some constructs defined in BMM are not seen in the BPMN. Such as *Business rules* or *policies* and objective, which is too specific to be on a more abstract business model.

BPMN and *i** are very close to each other in meaning. Most of the constructs defined in *i** models can directly be mapped to BPMN model. The best mapping between those two can be seen in *i** actors (*actor* Construction company) to BPMN pools and lanes (*pool* Construction company, *lane* Management), goals (*goal* rework complete) to states (*start event* warehouse rework completed), and tasks (*task* Ask retail chain to launch an online shopping system) to tasks (*task* Let the retail chain know about the online shopping). Again, more examples can be found in the mapping table (Appendix 1). Also, message and sequence flows can be descried as dependencies between the actors in *i** models. For example, *message flow* Funds between the management (*lane* management) and construction company

(*pool* Construction company) can be seen as *resource dependency* Funds in the *i** models (*actors* Retail chains management and Construction company).

Table 5. Similarities between BMM, BPMN, and *i** in constructs.

BMM	**BPMN**	***i****
Organization unit – Participates in defining and using the model	Pool – Container for the process	Actor – Someone who wants to achieve the goals
Mission – Ongoing activity to cover tactics and strategies	Start event – An event to start the process	Goal – Something to be achieved
Tactic – Something that is needed to be done to achieve the goals	Task – An activity to be done	Task – Specific way of doing something to achieve the goal
Not defined	Message flow – Flow between pools	Dependency – Depending on an another actor for something
Strategy – Something that is needed to be done to achieve the goals, implemented by tactics	Task – An activity to be done	Task – Specific way of doing something to achieve the goal
Goal – Narrow image of the end that is needed to be achieved	Intermediate event – An event happening in the middle of the process	Goal – Something to be achieved
Vision – Overall image of the end that will be achieved	End event – End of the process	Goal – Something to be achieved

6.4 Summary

In this Chapter the evaluations on language and model quality is presented, which based on the criteria defined in the Chapter 5. Also the observations of using BMM and *i** to make BPMN are discussed.

7 Discussion

This chapter will conclude this thesis. It consists of the limitations, related work, future work, and conclusion.

7.1 Limitations

It is needed to note that there are some threats to the validity in this work:

- *Subjective assessment.* The language and model evaluation is made by the author of this thesis. This may cause different results from a work done by other authors or bigger group of authors. To make this threat as minimal as possible, the criteria is chosen to be as objective as possible;
- *Few quality characteristics assessed.* This is a direct impact from *subjective assessment.* The criteria is chosen to be as objective as possible, causing there to be less criteria to be evaluated;
- *Limited versions of the languages.* Both languages, especially *i**, have various extensions, which are not taken into account in this thesis. If there would be various extensions included in the comparison, it may change the outcome;
- *Limited examples.* All the modelling examples are made to provide an example of the languages used. It contains most of the constructs defined in the language, but it is possible to make it even more specific and more detailed.

7.2 Related work

Goal-oriented modelling is getting more attention by time and because of that, there are multiple works done in the area that are similar in concept. An experiment done by Matulevičius and Heymans (2007) is carried out to compare two goal-oriented languages: *i** and KAOS (Knowledge Acquisition for Automated Specification). Some of the criteria used in that paper is also used in here. The evaluation is done differently, Matulevičius and Heymans made a questionnaire amongst the modellers to gather information. This causes differences in the comparison part, as the results from questionnaire will be subjective and without a concrete evidence on the topic.

Second work is represented in (Moody *et al.*, 2010), where the visuals of *i** are thoroughly analysed according to language notation design and making suggestions to the *i** visuals according to their findings. In this thesis, the analysis of visuals is in much lower in scope and just a part of the overall evaluation.

Third work observes the mapping between *i** and BPMN (Koliadis *et al.*, 2006). In their work, similar mapping between *i** and BPMN was noticed, *i** *actors* to BPMN *pool*, *i** *dependencies* to BPMN *message flows*, and *i** *tasks* to BPMN *tasks*. They also argue that the *i** *goals* map to BPMN *tasks*, while in this work, they are mostly mapped to *states* (*events*).

Fourth related work is similar to this thesis in concepts. It is a comparison between BMM and *i** (Tu, 2007). In his Master thesis Tu uses UEML (Unified Enterprise Modelling Language) approach, while here it is SEQUAL. Also, the work by Tu can be described as fine-grained comparison, meaning the comparison is in more depth but in smaller scale. This work is coarse-grained comparison, as there are many different criteria for the comparison and most of them are not in depth.

7.3 Conclusion

This work provides a comparison and evaluation of BMM and *i**, in both, language and modelling aspects. It also provides a mapping suggestion between BMM, *i** and BPMN models.

RQ1. The BMM and *i** have similarities regarding the ends and means. They both can be used to express phenomenon in goal and rule perspective. On one hand, the BMM means (tactic, strategy) and ends (goal, objective, vision) extend the meaning of *i** tasks and goals. On the other hand, the *i** language has defined dependencies between actors, that help to understand the organizational aspect better. This makes only the *i** language usable in actor and rule perspective.

RQ2. Both the models provided for BMM and *i** in this thesis are similar in quality, according to the criteria defined in Chapter 5. Small difference can be noted in the colours used criteria, where BMM is worse, as it exceeds the amount suggested. In the other hand, *i** is worse off for finding the goal of the model, as it is an internal element for one of the actors, while in BMM the *end* concept is dedicated to that. It should be noted, that it also depends on the tools used by the modeller.

RQ3. The *i** framework supports the making of BPMN better than BMM as seen in Section 6.3. It covers almost all the BPMN constructs (*tasks*, *states*, *pools*, and *lanes*) with its own (*tasks*, *goals*, and *actors*) and also provides rationale between the different *pools* and *lanes* with the *i* dependencies*.

7.4 Future work

The thesis opens several future research directions. Further research is needed to understand how to use the *best* features of both languages; for instance, the *i** framework could be expanded with the additional constructs for *means* and *ends* taken from the BMM language. This would also lead to understanding of the new mappings to reason on the process models. However, these study directions also require the fine-grained investigation of the language quality.

8 References

Ayala, C.P ., Cares, C., Carvallo, J.P ., Grau, G., Haya, M., Salazar, G., Franch, X., Mayol, E., Quer, C.: (2005). A Comparative Analysis of i*-based Agent-oriented Modeling Languages. In: Proceedings of the international workshop on agent-oriented software development methodology (AOSDM'2005).

Koliadis G., Vranesevic A., Bhuiyan M., Krishna A., Ghose A.: (2006). Combining i* and BPMN for Business Process Model Lifecycle Management. Business Process Management Workshops, pp 416-427.

Krogstie, J.: (2012). Model-Based Development and Evolution of Information Systems. London: Springer-Verlag.

Matulevičius R., Heymans P.: (2007). Comparing Goal Modelling Languages: an Experiment. Requirements Engineering: Foundation for Software Quality, pp 18-32.

Moody D. L., Heymans P., Matulevičius R.: (2010). Visual syntax does matter: improving the cognitive effectiveness of the i* visual notation. Requirements Engineering, 15, pp 141- 175.

Object Management Group: (2015). Business Motivation Model (BMM, version 1.3), URL: http://www.omg.org/spec/BMM/1.3/ (last check: 12.05.2016).

Tu, C.: (2007). Ontological evaluation of BMM and i* with the UEML approach. Master thesis, University of Namur.

Tõnisson R., Matulevičius R.: (2016). A Coarse-grained Comparison of Modelling Languages for Business Motivation and Intentional Distribution. Submitted for publication 2016.

Yu, E.: (1995). Modelling Strategic Relationships for Process Reengineering. Ph.D. thesis, University of Toronto.

Yu, E.: (1997). Towards Modelling and Reasoning Support for Early-phase Requirements Engineering. In: Proceedings of the 3rd IEEE international conference on requirements engineering (RE'97).

Appendix

I. Mapping between BMM, *i**, and BPMN.

BMM	BPMN	*i**
Organization unit [Retail chains management]	Lane [Management]	Actor [Retail chains management]
Deriving from tactic [Hire a construction company]	Pool [Construction company]	Actor[Construction company]
Deriving from tactics [Hire workers/delivery company]	Pool [Delivery and warehouse workers]	Actor [Delivery and warehouse workers]
Deriving from tactic [Hire a software engineers to build the system]	Lane [Engineers]	Actor [Software engineers]
Deriving from tactic [Test the software to be user friendly]	Lane [Testers]	Actor [Testers]
Assessment [Customers would like an online shopping system with fast delivery	Start event [Start shopping online]	Goal [Build the online shopping system with delivery]
Not defined	Task [Ask retail chain to launch an online shopping system]	Task [Let the retail chain know about the online shopping]
Not defined	Message flow [Customers wishes]	Resource dependency [Customers wishes]
Influencers [Warehouses need rework] and [Need an online shopping system]	Start event [Start online shopping]	Goal [Build the online shopping system with delivery]
Deriving from Tactic [Rework the warehouses for better delivery times]	Task [Give funds to rework the warehouses]	Task [Give funds to rework the warehouses]
Not defined	Message flow [Funds]	Resource dependency [Funds]
Not defined	Start event [Construction company hired]	Goal [Get funds]
Deriving from Tactic [Rework the warehouses for better delivery times]	Task [Plan the warehouse rework]	Task [Make plans for the rework]
Deriving from Tactic [Rework the warehouses for better delivery times]	Task [Rework the warehouses]	Task [Rework the warehouses]
Deriving from Tactic [Rework the warehouses for better delivery times]	Intermediate event [Warehouse rework completed]	Goal [Rework completed]
Not defined	Task [Hand over the warehouse]	Goal dependency [Reworked warehouses]
Not defined	Message flow [Warehouses]	Goal dependency [Reworked warehouses]
Deriving from Strategy [Make an easy to use system with fast delivery]	Intermediate event [Rework complete]	Goal [Warehouses reworked]
Deriving from Tactics [Hire new workers] and [Hire delivery company]	Task [Hire a delivery company and warehouse workers]	Task [Hire warehouse and delivery workers]
Not defined	Message flow [Contract]	Resource dependency [Contract]
Not defined	Start event [Workers hired]	Goal [Workers hired]
Tactic [Get the workers to be more efficient]	Task [Learn the delivery process]	Task [Learn the delivery process and packing process]

I. Mapping between BMM, *i**, and BPMN. (*Continued*)

BMM	BPMN	*i**
Deriving from Strategy [Make an easy to use system with fast delivery]	Intermediate event [Delivery system working]	Goal [Delivery system working]
Not defined	Task [Start the delivery system]	Goal dependency [Working delivery system]
Not defined	Message flow [Working delivery system]	Goal dependency [Working delivery system]
Deriving from Strategy [Make an easy to use system with fast delivery]	Intermediate event [Delivery working]	Goal [Working delivery system]
Task [Hire software engineers to build the system]	Task [Hire software engineers and testers]	Task [Hire software engineers and testers]
Not defined	Task [Give funds and requirements to build the system]	Tasks [Give funds to build the system] and [Give requirements to build the system]
Not defined	Message flow [Funds and Requirements]	Resource dependencies [Funds] and [System requirements]
Deriving from Task [Hire software engineers to build the system]	Task [Build the system]	Task [Build the system]
Not defined	Sequence flow [Online system]	Resource dependency [Online system]
Task [Test the system to be user friendly]	Task [Test the system]	Task [Test the system]
Not defined	Sequence flow [System tested]	Goal dependency [System tested]
Task [Hire software engineers to build the system]	Task [Fix the errors]	Task [Fix the errors]
Not defined	Sequence flow [Functional online shopping system]	Resource dependency [Functional online shopping system]
Deriving from Strategy [Make an easy to use system with fast delivery]	Task [Online shopping system built]	Task [Online shopping system built]
Deriving from Strategy [Make an easy to use system with fast delivery]	Intermediate event [System built]	Goal [Online shopping system built]
Goals [Have a working system …] and [Have a working delivery …]	Intermediate event [Functional shopping system with delivery]	Goal [Get a fully functional online shopping system with delivery]
Mission [Be able to buy groceries anywhere, at any time]	Task [Open the online shopping system with delivery for public use]	Task [Open the online shopping system for public use]
Not defined	Message flow [Online shopping system]	Resource dependency [Online shopping system]
Deriving from Vision [Online shopping and delivery working]	Task [Start buying goods online and getting them delivered]	Task [Start using the online shopping system]
Vision [Online shopping and delivery working]	End event [Online shopping started]	Goal [Buy groceries online and have them delivered]

II. License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Rando Tõnisson** (date of birth: 02.11.1993),

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:
    1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and
    1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis

Comparison of Business Motivation Model and Intentional Distribution,

supervised by Raimundas Matulevičius,

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **12.05.2016**