Introduction

The business analysis process has taken us from an understanding of the business context, defining the current state, analyzing the needs, capturing the outline of the future state, identified and evaluated alternative solutions, and finally, selected the main path forward (main solution). The solution is oftentimes described on a general level where the main features are captured in various documents and models. Naturally, there are variations in how detailed the current state, the future state, or the change strategy is captured depending on factors such as size, complexity, and urgency of the problem or initiative. Some might move into a project organization (deliver solution) at this point, other organizations might wish to have a better view of the financial aspects or simply want to have more detailed solutions before taking the final decision on initiating a project. Some organizations also take the decisions successively by deciding at each step, if to move forward with the next step, to cancel, or to revisit previous step. Furthermore, the project approach the company uses, also has an effect on this matter. Agile methods move into a project organization much sooner as compared to those companies that essentially are influenced by traditional waterfall approaches.

Regardless of the above factors, the solution is not defined in enough detail for construction (coding). Furthermore, while the solution is outlined in various degrees of detail, there are several possible ways it can be designed and these have not been clarified or any specific one chosen. As such, the solution is not ready, it needs to defined in more detail, the requirements need to be analyzed and captured at a deeper level, and the design must be set.

A note clarifying the distinction between requirement and design is in order. The analyst works with identifying and defining the needs and identify solutions. In the beginning phases of the analysis work, we focus on business needs on high level. As the work progresses, these needs are refined and expressed in more detail by being decomposed into "sub-needs" or several needs that together make up the main need. Note that once the needs or problems have been defined (assuming it is the “right” one), they do not change as the analysis process progresses. However, the understanding of how need can be satisfied or the problem be solved increases. As the analyst describes the ways the needs will be satisfied or the problem are to be solved, the descriptions become requirements.

Business needs expresses the “why” and business requirements states at a high level "what” is required to solve the issue. Oftentimes there are several alternatives to satisfy the need or in other words, fulfil the requirement. Each of these alternatives might have advantages and disadvantages. The selection of the alternative or in other words, selecting how the requirement are going to be fulfilled in more detail, is design. A word of caution is in place regarding the term "design”. In the IT domain, design is often used for describing technical designs. That is not the definition used in BABOK. With this
view, IT might consider every input given to them by the business as requirements. At the end of the day, different organizations might use different terms. That does not matter as much as the analyst being clear about what word refers to what concepts. The interplay between requirement and design is applicable at different levels but are called differently. At higher levels such as at strategic levels, the terms need and solutions are used. However, at more tactical levels where details matter, the terms requirement and design are used. Although different names are used, it is the same relation but at different levels of detail.

Requirement elicitation is an area that has been researched extensively. Many theories, methods, approaches, principles, and guidelines have been provided by industry and academia. Commercial programs have also been developed to support requirement elicitation and management. Although we see many a variety of methods and tools, they share a common core of concepts and ideas. These commonalities are more or less the underlying foundation of most methods. While the analyst might be proficient in one or perhaps two methods or tools, he or she should be open and able to adapt to working with other methods. Rather than briefly introducing different methods, we will focus on the foundational areas. This will help the analyst to understand and be able to use different methods faster and more efficiently.

When working with requirements and design, particularly at this stage of the analysis process, the main areas are as follows:

- Specify and Model Requirements, which is about using different analytical methods to capture, model, and describe requirements and designs.
- Verify and Validate Requirements i.e., ensuring that the requirements are at the appropriate level of detail and are of sufficiently high quality. Validate requirements or in other words, examine and ensure that the requirements and designs are aligned with the goals of the initiative and will bring desired value.
- Define solution options by identify, examine, explore, and describe alternative ways or design options to meet the requirements. The analyst will also analyze potential value and recommend solution by assessing and comparing the business value each design option will deliver so to recommend the best option.

In addition to the points briefly described above, we will also discuss two additional aspects related to designing solutions. The first is about eliciting requirements from business process models. Traditionally, requirement elicitation methods are born of IT people and predominantly have the IT perspective. However, the source of the requirements, oftentimes business people think in terms of workflows or processes. Business process models are therefore easier for business people to follow and have the business process perspective. This allows the business people to discuss requirements from the perspective of their processes rather than from an IT perspective they might be alien to. We cover this aspect as eliciting requirements on the basis of process models is usually not covered in standard elicitation courses. Furthermore, we will discuss managing the requirement life cycle. Although the requirement life cycle is not specifically part of designing the solution, it has an important relation this phase. Commonly, the number of requirements start growing and it will be important to
manage them efficiently. As mentioned before, there are tools but we will discuss the generalities of requirement life cycle management.

**Specify and Model Requirements**

The main purpose of specifying and modeling the requirements is simply to detail the requirements by analyze, refine, put together, synthetize and capture the requirements and designs. As input, the analyst has a fairly good view of the current state, the future state, the gaps, and the change strategy. The work now is to take those inputs and detail the solution.

In specifying and modeling the requirements, the analyst will predominantly work with choose the modeling format (including the level of abstractions and viewpoints), analyze, capture, and describe the requirements. Let us take a look at these elements.

**Choosing Models**

As requirements are elicited, analyzed, and discussed, they need to be captured or represented in some way. Oftentimes, textual description is simply not enough. Although textual descriptions can capture the requirements in detail, it will become difficult to find, understand, and work with a collective mass of requirements. As such, textual descriptions are better used for enhancing models by explaining and capturing important details. But the question remains regarding the choice of models. Some models focus on representing people, roles, and their relationships while others capture the data perspective of a particular domain or solution. The analyst will have to choose the models that are best suited for the context. Furthermore, it is always better to use models from different categories as each model presents the situation from a single viewpoint. However, problems or solutions are oftentimes complex and require several viewpoints in order to be captured adequately.

In BABOK, the different models have been categorized in the following way.

**People and Roles:** Models falling into this category focus on representing and depicting peoples, group of people, roles, organizational units, organizational entities, and their relationships with each other. These models view the problem or the solution from the perspective of stakeholders and roles but not much more. We have previously discussed some of the models in this category such as models capturing organizational structures, and stakeholders (stakeholder wheel or onion, and the RACI matrix).

**Rationale:** Models that capture the rationale of why an initiative is being considered or conducted, share the commonality of trying to analyze the question of “why” a change is needed. A few examples of such models are different problem analysis models, scope analysis, and business model canvas. The problem analysis models such as fishbone diagram and five why method aim at understanding the root cause of the problem and as such take a closer look to analyze the reason or the “why” a certain initiative is being
investigated. In similar manner, scope modeling tries to define the boundaries of the issue. Although it does not directly address the “why”, it aims at giving the problem or the solution some boundaries.

**Activity Flow:** Models that capture the flow or sequence of tasks, activities, or events in some form such as process models, use cases, scenarios or user stories, fall into this category. Such models are essential and should almost always be included. They describe how things are done and any changes will affect such flows and therefore, should be modelled. Furthermore, such models are more intuitively easy to understand by end users and non-IT resources.

**Capability:** Models that take the viewpoint of capabilities focus on what an organization can and does offer. We have previously discussed capability models that capture exactly that, the capabilities of an organization. Business model canvas and analysis also capture the capability of an organization but from a more holistic viewpoint.

**Data and Information:** The final set of models share the commonality of focusing on the data and data flow perspective of a particular problem or solution. These models are most often closely connected to information systems and have been influenced by that domain. Examples of such models are entity relationship diagram, data flow diagrams, and various models capturing the IT structure such as the interface models.

Oftentimes, models present the captured information either as matrices or as diagrams. Matrices have the benefit of being able to manage larger data sets that have some form of uniform structure. For instance, a data dictionary can present a large number of definitions and as each definition follows a uniform structure, it can be presented as a table (matrix format). However, if a data dictionary was to be presented as a diagram, it would not be able to contain more than perhaps a handful of definitions. In addition, it is possible to capture more data with a matrix format as compared to diagrams. For instance, when capturing a gap analysis, it is possible to add additional columns that contains valuable information allow for visualizing prioritization in a structured way.

Diagrams, on the other hand, expresses the information visually. When the complexity of the problem or the solution is high, it is difficult to capture it properly with just text or matrices. Much will be lost and it is difficult to get an overview of the situation. However, visual expressions of such cases allow for a simplification that captures the complexity in a manner in an understandable manner. For instance, if an entity relationship diagram was to be expressed in words, it would be very difficult to get a grasp of all the relationships. A model allows for an overview in a way words would not be able to capture. Most of the models we have discussed present information as diagrams of different forms, such as process diagrams, models of scope, and relationships between organizational structures, data objects, or IT systems.
Model and Analyze Requirements

Oftentimes, models have implicitly been selected during the current state analysis. It makes sense to re-use the same models. However, those models and requirements will most likely be at a too high level and therefore, need to be captured in more detail. It is fully possible that additional models will be chosen at this stage or not continuing with models used during previous stages. For instance, the business model canvas has more usefulness and value at a higher level when the contexts and perhaps needs are being explored and discussed. As such, the canvas will most likely not be valuable when detailing requirements.

One of the most time-consuming parts of the work of a business analyst is the detailed elicitation, specification, and modeling of requirements. However, the modeling and analyzing work is not just about collecting and documenting requirements. At this stage the analyst must take a closer look to see if any of the requirements or parts of a requirement needs to stay, be changed, modified, or deleted in order to deliver the intended value. Furthermore, when dealing with this level of detail, missing components can be identified and added accordingly. At the same time, there might be parts or full requirements that can be removed without limiting the desired value delivery. Finally, some requirements might need additional inputs, impose restrictions on other requirements, or work under certain assumptions. These factors need to be elicited as if they are overlooked, it might cause expensive re-iterations later.

Fortunately, there are quite a good number of tools for modeling and managing requirements. Most of the models we have discussed have tools or templates to assist the analyst. We noted previously that at this stage, the number of requirements grow and it will be necessary to have a structured method to keep the requirements organized. To this end, we will discuss the principles of requirement life cycle management but it should be noted that there are tools to help the analyst. For smaller projects, it is possible to keep a track of the requirements but when the projects get larger, a tool is highly recommended. Although there are many tools available, it seems that the most common way companies capture and work with requirements, is still word processing tools such as Microsoft Word.

As the requirements grow in numbers and start covering different areas of the solution, it will increasingly become important to ensure that the requirements are free from internal inconsistencies and conflicts, and that the collection as a whole will create the desired values. This is referred to as defining the requirement architecture. The main purpose of this part of the analysis is to ensure that the requirements will collectively deliver the desired value. Another important aspect is to better understand how requirements in one part of the solution affect or has dependencies to requirements in another part. It might happen that there are trade-offs that needs to be made and a requirement architecture can support in making good decisions in such cases.
At this stage, the stakeholders are still very relevant and important to ensure that the right solution is being built. The devil is in the details. The requirements need to be approved by relevant stakeholders. As such, how the requirements are presented matters. Although they are at a more detailed level, they still need to be presented to various stakeholders. However, not every stakeholder will require or even find it valuable or understandable if presented with detailed requirement specifications. Therefore, it will be necessary to capture and represent the requirements at different levels of abstraction and from different viewpoints depending on who the stakeholder is and what their particular expertise and interests are.

**Verify and Validate Requirements**

Verifying and validating requirements concern the quality aspect of requirements. Verify is about ensuring that the requirements have are captured in a way so they can be used whereas validate focuses on ensuring that they deliver value.

From a quality perspective, requirements can be irrelevant, inadequate, or bad. Some requirements might be captured and illustrated in a beautiful way, in other words, they are very good but simply irrelevant.

Irrelevant requirements, while not being “wrong”, are those that are redundant. They simply do not help clarifying the solution, they cannot be tested, or are out of scope. Such requirements can creep in and due to their irrelevancy, working with them is wasted time. Bad requirements are those that for some reason do not provide the information needed. We will discuss criteria for good requirements in a moment. Finally, we have missing requirements. When projects grow in size or complexity, it can become challenging to cover all requirements and therefore, requirements go missing.

**Verify Requirements**

Verification of requirements focuses on ensuring the definition of requirement being adequate. Let us take a closer look at a few criteria.

**Understandability**: One must bear in mind that requirements are written in such a way that it is aligned with its intended audience. Imagine a set of detailed requirements intended for high-level executives who are to decide about the investment. Will these “do the work” or will the executives be lost in the requirements? Another source of confusion might be the choice of notation. Executives might not be accustomed to certain notations and therefore, would not easily understand for instance entity relationship diagram notation. Using such a notation will certainly not help in conveying the message. Requirements are meant to convey information in an easily understandable manner to an intended audience. As such, an aspect of the quality of the requirement concerns
its fitness for use. A set of requirements might be perfect for executives but completely inadequate for software coders.

**Completeness**: Another aspect to consider is if the requirements convey or hold all the information needed for further work. Completeness, therefore, depends on what the next step is. Certain models, methods or perspectives have their limitations or intended purpose which is reflected in the degree of details they cover. As such, a model might be considered as complete and fully adequate for further work whereas another model in the same situation is incomplete. Furthermore, as completeness considers further work to be performed, completeness is also related to the stage or phase of the analysis process. A requirement might be complete at an early stage but incomplete when it comes to design phase of the process.

**Atomic**: A requirement should be “self-contained” by which is meant that one should be able to understand it without having to refer to other requirements. Ideally, each requirement should cover one functionality, describing accurately what is going to be delivered.

**Concise and unambiguous**: A good requirement is concise, including only the information needed and therefore, free from all unnecessary descriptions, content or aspects. The conciseness also helps make the requirement clear. Unambiguity refers to the requirement being clear so that there is only one interpretation of what it means. Clarity is therefore that multiple persons who read the same requirement will understand and interpret in the same way.

**Feasible**: The requirement should also be feasible to do. Usually, there are restrictions in risk, budget, and/or time that needs to be considered. A good requirement considers these restrictions so to ensure they are feasible.

**Testable**: A good requirement is captured in such a manner as to make it possible to actually test it and see if it has been successfully implemented and the requirement fulfilled.

The above criteria are a good start to ensure requirements are of sufficient quality. It should be noted that the above criteria concern individual requirements. However, requirements seldom come alone. The set of requirements matters. As such, it is worth taking a quick look at some criteria for ensuring that the set of requirements is of adequate level of quality.

**Prioritized**: The set of requirements (and individual ones within the set) should have a priority indicating how important it is to have the requirement(s) implemented. The prioritization will depend on different factors but it is important to have a good view of the relative importance of the requirements.

**Consistent**: The individual requirements will, although each one might be perfect by themselves, be of limited value if they conflict other requirements.
The consistency should exist from high-level business requirements to low-level system requirements.

**Traceable:** An aspect of good requirement set, is that it is possible to trace (both the set and the individual requirements of the set) to its higher level and lower level requirements. This also touches upon the ability to modify the requirements and being able to consider what other requirements will be affected.

The actual verification of requirements does not happen as a one-time event. Rather, it is a continuous and iterative process running concurrently with the requirement elicitation. In other words, it is done in parallel as the requirements are elicited and models created. For instance, an analyst might work with modeling a process. In this work, the analyst might check the policies of the company to see what notation is preferred or required. In the first instance, the analyst is gathering the data in a workshop. As the process model evolves, the analyst ensures that the basics of the process model follow the notation but the focus is on the flow, not the notation. As the model is taking shape, the analyst asks questions to fill in the gaps or parts that seem to be missing information. After the workshop, the analyst might sit down and model the process with a process modeling tool. As the model is generated, the modeling tool might inform when the analyst tries to model in a way that is not aligned with the notation being used. As the model develops, additional questions might arise, which the analyst seeks the answer for and uses the information to make model complete. After the model is created, perhaps another analyst will take a look and comment. Perhaps the analyst will check if the model is consistent with other processes, particularly those preceding or coming after this particular one. The stakeholders might take a look and those who will work with the next step might be consulted to see if anything is lacking.

As can be seen, verification is continuously done. In essence, it is about ensuring that the models and requirements are of an adequate quality. That might include activities such as ensuring compliance with corporate policies, standards, and guidelines for tools, models, and approaches, continuously checking for missing parts, completeness, comparing with other models to capture inconsistencies, reviewing to make sure the right terms are being used, and perhaps add explanations and examples for the purpose of clarification.

**Validate Requirements**

The requirements define what is going to be delivered. It is therefore important to ensure that the functionality that is to be delivered is the “right” one for the solution or in other words, to validate the requirements. Note that validation is different from verification. Verification serves to ensure that the requirements are stated correctly whereas validation ensures that the right requirement is stated. In other words, verification checks for the quality of the requirement specification and validation checks if the requirement is useful for the business. Validation is, according to BABOK, to
“ensure that all requirements and design align to the business requirements and support the delivery of needed value”.

The business analyst has a vital role in securing that the intended solution actually will satisfy the business needs and resolve the problems that motivated the initiation of the solution. In order to validate requirements, the analyst needs to refer back to the business needs, the problems defined and the future state. In essence the analyst needs to secure that the requirements will bring the organization to the defined future state.

There are two main questions or aspects that can be used as basis for requirement validation (functional and non-functional). These are as follows.

1. Do the requirements bring business value?
2. Do the requirements satisfy stakeholder needs?

Validating a large number of atomic requirements is very difficult and time-consuming. It is more feasible to validate groups of requirements. Most requirement elicitation methods include a structured way of documenting the requirements. For instance, a set of requirements might be grouped under a capability such as automated check of orders. It is more feasible to validate these requirements by first validating the capability and then each requirement within that group.

**Business Value**

The requirements should express what is to be developed to realize the defined future state. Validating the requirements is therefore very close to validating that the future will be achieved. The validation aims at ensuring the delivery of, for instance an automated check of orders. Some questions might be helpful in determining this.

1. What assumptions are made for each requirement?
2. Does every requirement add value for the solution (i.e., do they all play a necessary role for enabling automated check of orders)?
3. Are any requirements good to have or desirable but not necessary for an automated check of order?

The future state should include metrics expressing the improvements in numbers. For instance, in the current state 100 orders can be checked per day by 4 persons. However, for the future state, the need is to have automated check that enables 1000 orders to be checked and have 1 person dealing with faulty orders. Ensuring that the requirements satisfy this need is also part of the validation. This part of the validation can be conducted by examining the details of the solution to ensure that there are no bottlenecks, dependencies that can cause limitations in the number of orders checked.

The requirements are validated against these criteria defined in the future state and the target metrics. Oftentimes software solution requirements are validated. It might be difficult to view such requirements in light of business value. However, the software is merely a tool to achieve improvements and not the improvement itself. Therefore, such
requirements are viewed as parts of the tool and validated in terms of if and how well they will realize the improvements (and its metrics).

The assumptions can play a very important role. If an assumption for the requirement states a dependency on another requirement, solution or project, the whole solution might fall short of delivering its intended value. The validation ensures that the assumptions made together with the requirements defined deliver a solution that meets the need or resolves the problem.

Stakeholder Needs

Once the requirements are validated in terms of business value, it is important to consider if the stakeholder needs will be satisfied. Different stakeholders might have different interests and perhaps even conflicting views. The same solution might meet a business need that is shared by two or more stakeholders but each stakeholder might have a different need.

For instance, two stakeholders might need the automated check of orders but differ in their interest. One might wish to have it to reduce costs by reducing number of persons in the back office whereas the other might want to increase the number of orders as more sales would take place (sales). Both have a need to increase the number of orders checked but from different perspectives. The solution must not only satisfy the need of back office but also of the sales department. The back-office will have interest in terms of how incorrect orders are managed and how they can quickly be corrected but sales might focus on the contents of the orders. Validation of requirements is therefore also ensuring that all stakeholders receive a solution that satisfies their distinct needs.

Benefit network analysis

The analyst should have a good view of which requirement what value and what business need it satisfies. However, this is not always easy. This is also particularly important if there are changes and the implication or the effects of the changes needs to be assessed. A tool that might be helpful is a benefit dependency network (henceforth BDN).

BDN was initially constructed to visualize the relation between IT investments and the benefit they hoped to deliver. Although BDN was initially considered for business to manage their IT portfolios (several IT investments), it is fully possible to use it for one specific IT investment. Regardless, in capturing the relation between IT investments and the benefits, there are seven questions BDN aims at answering. These are as follows.

- Why is it important for the company to improve?
- Which improvements are necessary or possible?
- What benefits can be expected (for each stakeholder) and how can the benefits be measured?
• Who is the owner of the benefit that ensures the its realization?
• What needs to be changed in the current state to enable the benefit to be achieved?
• Who will be responsible for ensuring the implementation of the changes required that deliver the benefit?
• When (and possibly how) can the identified changes be implemented?

As can be seen from the above "questions", we see that BDN can be of assistance to the business analyst. Benefit Dependency Network (BDN) can therefore, be helpful for understanding the connection between different parts of the project (which includes the requirements) and the expected benefits of the project. BDN is also useful as an overview when discussing changes in requirements with stakeholders. The BDN explicitly connects and depicts the essential IT functionalities that enable a change in the business, which will produce benefits that ultimately achieve the overall investment objectives.

The BDN has five components. These are as follows.

• Objectives – the end result or the effect that an organization seeks to achieve. For instance, it can be "reduce operational costs" or "increase sales volume".
• Benefits – the benefits that will lead to achieving the objectives. For instance, to reach the objective of "reduce operational costs", the benefits can be "reduced costs from less faulty products" and "higher volume per person managed".
• Business Changes – the long-lasting changes introduced to for instance business processes, operations, practices that will cause the benefits. Normally such changes come once a new solution has been implemented. For instance, a business change of using an information system to track orders can only be used once the system has been implemented.
• Enabling Changes – the predominantly one-time changes that are made or introduced that are essential for making the changes in the business (business change) such as business process re-designs or implementation of a new customer relationship management system.
• IT Enablers – the IT components/solutions/elements that are needed for enabling the changes required.

In building a BDN, the objectives of the investment need and the benefits that the IT investment can deliver must be discussed and agreed upon. These should be clear at this stage of the analysis process. Following this, the "business changes" are identified. These are oftentimes permanent changes must be made alongside of after the IT investment is deployed. This could be for instance new capabilities, new processes, or changes in roles. Then, the prerequisites for making the "business changes" possible need to be identified. These are called "enabling changes". This should sound somewhat familiar as they were discussed in "change strategy" and in particular, the gap analysis. Finally, the “IT enablers” or in other words, the actual functionalities of the project are identified. Once these results are captured (see figure below), they are illustrated with the IT enablers to the further most left and successively building the relation of each IT enabler until reaching the objectives (placed at the further most right).
In the simple example above, we see that the company seeks two main objectives, namely to grow by getting new customers and by growing existing sales. This is the reason why they seek to make an IT investment. We also see that the benefits of the project will deliver. One benefit is reduction of costs associated with spending time on bad leads and by doing so, also not spend time on good leads of after sales. Another benefit is to target new customer segments and thirdly, to refine existing customer segments to be able to make more targeted campaigns with better results.

Moving to the other side of the diagram, we see that the main functionalities will be data mining algorithms, a new customer database (perhaps with more refined structure and able to capture more data), and online access to the IT system so sales personnel can access and do the work from remote locations such as when visiting customers. We can also see that online access (registry of contracts) enables new sales processes and data mining algorithms together with the new customer database will enable better filtering of leads. This enabling changes will allow the company to make a more permanent change of increase the time spent on quality leads, aftersales and existing customers. These changes will in turn, yield the benefits listed above.

As the project is initiated and progresses, more data become known and can be used. The new data can be used to enrich and complement the BDN. The BDN depicted above is an illustration and can be modified to fit specific needs. For larger projects, it might be feasible to use several BDN illustrations, each for every major component of the project.

BDN can be helpful for the business analyst to secure that the objectives of the IT investment are aligned with the business objectives. The structure of a BDN helps to trace what functionality is connected or enables what objectives. Furthermore, BDN can help create a common understanding and therefore, agreement among different stakeholders on various aspects. Finally, as mentioned, the BDN provides an overview of
Define Design Options and Recommend Solution

There are usually alternative ways of solving a problem or fulfilling a requirement. If an analyst goes with the first one that come to mind, there is a risk of overlooking a better design. The idea of defining design options is therefore to see what options there are and then choose. Commonly, as the solution starts to evolve, many aspects surface and different ways to achieve the goals emerge. However, each alternative will have its benefits, strengths, advantages, and disadvantages. Contrasting these options with each other, allows for better choices.

When working with defining options, be it of higher level where approaches are discussed or at lower levels of different alternative solutions), improvement opportunities emerge. In general, when different options are being discussed, new aspects are brought to our attention, some of which are better than the ones we initially considered. Some might be conducive to improve efficiencies by for instance enabling some processes to be performed faster or cheaper. An alternative option might enable a change in a process that reduces time, another alternative might enable outsourcing of some parts. Other solutions might enable better distribution of information among co-workers or enable incorporation of new capabilities that can be used at later stages to enhance the business. The main point is that alternative solutions often provide opportunities to improve the solution a bit more.

Once the alternatives are identified, they can be analyzed. It is worth bearing in mind that oftentimes, these alternatives are not formally elicited as a separate activity. Quite often, as discussions are being made about a specific part of the solution, ideas will be exchanged and discussed. Gradually, as more and more information is gathered and the matter discussed, one or two main alternatives emerge as the most feasible and best options. Oftentimes, deciding between these is quite straightforward but occasionally, the trade-off is significant enough to merit further discussions. Besides the actual designs, it is important to consider the expected costs, benefits, and risks of the alternatives. The cost manifest itself as possible delay it timeline, efforts required, operating and maintenance costs, external costs (if purchases are required), physical, human, or informational resources required. The value on the other hand, is perhaps more difficult to estimate. However, care should be given not to equate value versus
potential value. While potential value can be important, it is uncertain value and as such, should be given less emphasis as compared to actual value that will be realized.

When recommending an alternative, the best one is not always the one to move forward with. Before making the final recommendation, it is important to consider the available resources. If there are any limitations in resources that affect the ability to implement a specific alternative, that should be taken into consideration. Furthermore, a holistic perspective must be considered. It might happen that several alternatives do not bring much value but are required for delivering another functionality that is vital. As such, the dependencies to other parts might matter when recommending an alternative.

As with verification and validation, these aspects are not one-time events but take place alongside the work. Stakeholders are vital in this process but the analyst can eliminate alternatives that are out of scope, unreasonable, or contradicting other important aspects. In this way, the decision is made easier.

**Requirement Elicitation using Business Process Models**

**Introduction**

In organizations, process models are used to facilitate communication between various stakeholders, to understand how work is being performed and where improvements can be made. Such process models are valuable sources of information for requirements elicitation. In fact, these models are not only used to understand the environment but are increasingly becoming an important part of the requirements specification process. Business process models, while being widely used, are rarely utilized as the main artifact when discussing requirement with domain experts. In the following sub-sections a systematic method for eliciting requirements from business people by using process models as common artifact is presented. The method assumes that the to-be business process has been modeled and can be used as basis for the elicitation. The method provides a template, which includes the data needed for a requirement, and a set of questions that will guide the elicitation of requirements in collaborative discussions, based on business process models, with the domain experts. For each relevant activity, questions are asked of the domain experts that allows for eliciting the intended requirement.

**Elicitation Process**

A template is used to document the requirements. The template is filled for every relevant activity of the process model. This means that in principle, for each activity of the process, there is one main requirement. The template covers all aspects that constitute a complete requirement. The template is populated by asking a set of pre-defined questions from the domain experts while using the business process model as main artifact.
The process of populating the requirement specification is practically achieved by eliciting information about the goal, actor, trigger, operational steps, alternative paths, failure conditions and their management for each relevant activity (see figure below). The information required, is elicited by applying a set of questions that are designed to capture that specific information from the domain experts using business process models as common artifact. As such, the requirement specification template is gradually defined until it forms a complete specification. The process is then repeated for each activity of the business process that is relevant.

Figure 2 The Requirement Elicitation Process

The extent, to which the domain experts are engaged, depends on the level of detail in the business process models. If the process models have been modeled in great detail, most of the information is already captured in the models. In such cases, the input of the domain experts is of a more confirmatory nature. However, if the models are not of such detailed level, “hidden” requirements are elicited from the domain experts. For instance, if the model lacks artifacts, the elicitation will inquire about the objects and capture, through the questions, the information from the domain experts. By the same principle, incomplete models can be made complete by adding the lacking parts to the model.

Table 1 Requirement Specification Template

<table>
<thead>
<tr>
<th>Requirement Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID:</td>
</tr>
<tr>
<td>Business Process:</td>
</tr>
<tr>
<td>Activity:</td>
</tr>
<tr>
<td>Goal:</td>
</tr>
<tr>
<td>Actor:</td>
</tr>
<tr>
<td>Trigger:</td>
</tr>
<tr>
<td>Steps of Activity (positive scenario)</td>
</tr>
<tr>
<td>Alternative paths:</td>
</tr>
<tr>
<td>Failure conditions and management:</td>
</tr>
</tbody>
</table>
Activities are the focal point of the method. Each requirement specification corresponds to at least one activity. The requirement specification [see Table 2] consists of two columns, where the first one states what data is to be captured in the second column of each row. The data required are "id" (a unique id for the requirement specification), "business process" (name of the process model in which the main activity of the requirement specification belongs to), "activity" (the name of the focal activity that is the object of requirement elicitation), "goal" (the expected outcome of the activity), "actor" (the performer of the activity), "trigger" (what initiates the actor to perform the activity), "procedures of the activity" (the operational steps taken to perform the activity, both desired steps and alternative steps required when the desired steps cannot be executed), and "failure conditions and handling" (cases where the activity cannot be executed or interrupted in its execution and actions to handle the failures).

The requirement specification template, which is inspired by use case specification of Cockburn, covers all components necessary for a requirement as aligned with their corresponding business process model elements.

**Step 1 – Determine Relevant Activities**

The first step is to determine if the activity is relevant or not. An activity is considered as relevant if it requires some form of functionality support from an information system. If the activity is performed manually and does not require any support from any semi-automated or automated system, it is not relevant. As such, a specification is only elicited for activities that require some sort of support to be considered or developed.

A requirement specification is populated for each activity. However, in some cases, several activities are so connected that they should be treated as one from the perspective of requirement specification. To determine if two or more adjacent activities should be included in one requirement specification, the following questions as inspired by Cockburn are asked: (1) Are the consecutive activities executed by one person, in one place and at the same time? (2) Is it possible or reasonable to have a break between the activities? If the answers to both questions are "yes", the execution of the activities are tightly connected and there is no reasonable reason to separate them. They should therefore, be treated as one requirement for the system being built.

**Step 2 – Identify Relevancy of Activity**

The second step is to determine if the activity is relevant. If the activity requires some form of system support, there is a need for having its functional requirements specified. In order to determine the relevancy of the activity, the following questions are asked:

- Does the execution of the activity require any support from any computer-based system?
• Is the system under construction to be involved by providing, executing or receiving data during the execution of the activity?

• Are there any connections to external systems involved in the execution of the activity that need to be considered for interfacing with the system under construction?

Step 3 – Elicitation of Goal

An executed activity serves to fulfill a certain predefined outcome or goal. In this step, the outcome of the activity is elicited by asking the following questions:

• What changes after the activity has been executed?

• What is required to be achieved or accomplished with the execution of the activity?

• In what form and/or format are the results in?

Step 4 – Elicitation of Actor

In this step, the executor of the activity is elicited. The actor can be either human such as a role, department or organizational unit or a non-human resource such as an information system. The actors are elicited by asking the following question:

• Who are the actors, human and non-human, who are involved in the execution of the activity?

Step 5 – Elicitation of Trigger

Triggers determine when an activity is to be executed. Activities are generally triggered by either an actor receiving a message, a specific predefined time or by the end of a preceding activity. The following questions assist in eliciting the triggers:

• How does the actor (human or non-human) know when to start the execution of an activity?

• If it is a message, what kind of message is it and how does the actor become aware of receipt of the message?

• If the activity is time-dependent event, how is the actor notified about when to start the execution of the activity?

If it is complete execution of the preceding activity that is the trigger, is the actor responsible for the execution of the preceding activity and if not, how is the actor informed about it?
Step 6 – Elicitation of Operational Steps

An activity usually consists of procedural or operational steps, i.e. the individual steps that need to be carried out in order to execute the activity. In this step, the preferred or desired operational steps are elicited by asking the following questions:

- What are the operational steps required for the execution of the activity?
- Who performs the operational steps?
- What tools or aids does the actor engage or use in carrying out the operational steps (such as human or non-human actors, internal or external systems)
- How are these tools or aids used?
- Are verifications required in carrying out the operational steps?

Step 7 – Elicitation of Alternative Paths

Alongside the standard set of operational steps, there are alternative paths taken when the standard cannot be executed. This could be for instance, entering an order when the customer is not registered and an alternative path is required before the order can be registered. These alternative paths are elicited by asking the following questions:

- Are there cases (when carrying out the standard operation steps) where additional or alternative steps need to be taken in order to reach the goal of the activity?
- What are the conditions of these cases?
- What complementary or replacing steps need to take place in such cases?

Step 8 – Elicitation of Failure Conditions and Failure Management

Activities cannot always successfully be executed and reach its goal as they might be interrupted or disrupted. In this step, such conditions that hinder an activity from being initiated, interrupted or disrupted are elicited. Furthermore, such failure situations require additional steps to be taken in order to solve the disruption. These failures and steps to manage them are elicited with the aid of the following questions:

- What can hinder the initiation of an activity?
- What can cause to interrupt or disrupt an activity?
- What activities or steps are needed to limit the loss, handle or resolve issues so an activity can be initiated?

Example of Check and Update Order Confirmation

In the first step, the activity, "check and update order confirmation" was determined as relevant because it requires some interaction with a system support (not a purely
manual task). Then, the goal of the activity is elicited (step 3). With the aid of the questions, it becomes clear that the goal of this activity is to achieve an updated order. After this, the actor is determined (step 4) which is someone from the purchasing department. The next set of questions aim at eliciting the trigger of the activity (step 5). In this case, a message event preceded the activity indicating that an incoming message from the supplier is the trigger. This is further clarified (with the aid of the questions) that the trigger is an email with an attachment from the supplier. Further discussion reveals that there is no need for any automation or an interface. Next, the operational step of the activity is elicited (step 6). By using the questions, the operational steps were elicited and clarified. Some steps, such as the second step, “find the relevant purchase order” are elaborated as to what parameters are used to find an order. Following the operational steps, the alternative paths (step 7) are elicited. The discussions based on the questions of the method, reveals that two alternative paths exist, one for when the confirmation differs from the order and when the suggested delivery date is later than the customer needs the goods. The final step of the method (step 8) is eliciting failure conditions and management. In this step, situations that prevent the activity from starting or that interrupts/disrupts the activity and the measures needed to be taken are discussed. Naturally, the failures are connected with the operational steps and alternative paths. For instance, the alternative path of order confirmation differs from the order and it is not acceptable, the management of this situation is clarified here. In this case, it is to delete the order and the process is interrupted.

Table 3: Example of a Populated Requirement Specification

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID:</td>
<td>003</td>
</tr>
<tr>
<td>Business Process:</td>
<td>Supply chain security (purchase)</td>
</tr>
<tr>
<td>Activity:</td>
<td>Check and update order confirmation</td>
</tr>
<tr>
<td>Goal:</td>
<td>Updated order (suggested delivery date and order status updated)</td>
</tr>
<tr>
<td>Primary Actor:</td>
<td>Purchase department</td>
</tr>
<tr>
<td>Trigger:</td>
<td>Order confirmation received by e-mail</td>
</tr>
<tr>
<td>Steps of Activity</td>
<td><strong>Operational steps:</strong></td>
</tr>
<tr>
<td>(positive scenario)</td>
<td>1. Open PDF format order confirmation received by email</td>
</tr>
<tr>
<td></td>
<td>2. Find the relevant purchase order</td>
</tr>
<tr>
<td></td>
<td>3. Check that ordered materials are the same as on the order</td>
</tr>
<tr>
<td></td>
<td>4. Enter suggested delivery date and change the status to</td>
</tr>
<tr>
<td>&quot;Confirmed&quot;</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>5. Reply the email confirming the order confirmation</td>
<td></td>
</tr>
<tr>
<td>6. Save the order</td>
<td></td>
</tr>
</tbody>
</table>

**Alternative paths:**

1. If order confirmation differs from the order (e.g. quantity smaller than ordered), contact the person who created the order and ask for advice; If changes OK follow the normal flow.

2. If suggested delivery date is later than the needed delivery date, take same actions as in alternative path 1.

<table>
<thead>
<tr>
<th>Failure conditions and management:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. If order confirmation differs from the order and is not acceptable, the order will be deleted and the process will be interrupted.</td>
</tr>
</tbody>
</table>

**Non-functional requirements**

We have discussed the designing of alternative solutions above but we have exclusively considered functional requirements. Most cases also require a closer examination of non-functional requirements as well. Functional requirements define how the system or the solution should something, in other words, the focus is on specific behaviors. Non-functional requirements, on the other hand, look at how well the functional requirements should perform.

A solution might have a set of functional requirements stating how a new application is to be processed, what data fields are to be filled in, which of these are mandatory and which are optional, how the solution should behave if wrong data is entered and so on. However, such requirements are not stating anything about how well the system should perform such as the system must be available and running 98% of the time. The solution might hold sensitive data so there might be non-functional requirements stating the level of security. In essence, non-functional requirements states characteristics a system must have, not what it must do.

There are some different categorization frameworks for non-functional requirements and while each have their specific approach, most list a core set of categories. One of the categorizations split the non-functional requirements into three main categories, product, organizational, and external requirements. Each is then decomposed further as shown in the figure below.
The perhaps most common categories of non-functional requirements are as follows.

- **Availability** - stating requirements on the time the system is expected to be available for use.
- **Maintainability** - expressing the requirements regarding level of ease by which components can for instance be changed, replaced, removed, or adapted to other systems.
- **Reliability** - defining how much the system is expected to be up and running without disruptions. Reliability is often expressed as for instance mean time between failures or ratio of down-time to availability.
- **Performance** - stating the volume, speed or another criterion expressing how well the system must perform. It can be expressed as for instance, the speed by which something must be completed or the capacity the system must manage at peak time.
- **Security** - expressing aspects concerning protection against intentional or accidental unauthorized access, usage, changes, or disclosure.

The above list is far from complete but captures the most common examples and occurring non-functional requirements.

The analyst might find it challenging to capture how to measure non-functional requirements. It is all too easy to express a non-functional requirement in vague terms. For instance, performance might be expressed as “the system must be able to manage the load at peak-time”. This is vague and will not help developers, nor will it help the analyst in assessing if the requirement is fulfilled. Rather, it must be made more quantifiable. For instance, if the record peak has been 100 000 transactions, the statement “the system must be able to manage 120 000 transaction per day”, is better. However, this might also be misleading. If the peak is evenly spread over one day, then the above statement is in order. However, if it happens that most of the transactions...
come within a few minutes and the rest are evenly distributed over the day, then the statement is misleading. In fact, the system might be able to manage 120 000 transactions per day but break down on a day when the total transactions were 50 000 transactions. In such a case, the statement, “the system must be able to manage 60 000 transactions within 20 minutes” is more appropriate. As can be discerned, defining non-functional requirements require the analyst to have some foundational data to understand what is expected of the system and define the non-functional requirements accordingly.

**Vendor Assessment**

To be written later. Include “vendor assessment” from BABOK, SaaS vs Inhouse from SoftEco slides, and Assessing Vendor solutions based on training course in Riga.

**Feasibility Study**

At this stage of the work, the solution has been defined. The level of detail in regards to the requirements vary. Some prefer to keep it at a higher level while others see value in detailing the solution a bit more. The level of detail is also influenced by the project approach such as waterfall or agile methods. When the results of the business analysis work do not go into the details of the solution, it is oftentimes referred to as a “business requirement document”. The requirements are not supposed to go into too much detail as that will be the responsibility of the project. However, at the end of this stage (design solution) everything should be ready for a final decision and project start. Again, this will depend on the project method. Some organizations operate in an industry where one method is preferable to the other. Furthermore, internal policies also play a role in how detailed the requirements should be prior to a decision is taken to proceed or not. It is equally possible to produce the business requirement document before this stage (design solution). There are no rules and as such, the analyst should be flexible. Regardless, the results of the work so far need to be documented. Sometimes all the results are captured in a document called “feasibility study”. To give an overview, below is a suggestion of topics that could be included in a feasibility study.

1. Executive summary
2. Recommendation
3. Background, scope and delimitations
4. Current State
   a. Stakeholder analysis
   b. Business process models
   c. Information system map (data flows, interfaces and functionalities)
5. Business needs and problem analysis
6. Future State
7. Alternative solutions
8. Costs and benefits
   a. Cost estimation
   b. Benefit analysis
   c. Financial analysis
   d. Risk analysis
   e. Readiness analysis
9. Selected alternative (including motivation)
10. Results from validation of requirements

A feasibility study might also cover the main inputs for the project as well. If that is to be included, it traditionally includes the following other aspects.

   a. Dependencies
   b. Stakeholders affected (and how)
   c. Critical factors
   d. Gained approvals
   e. Delivery (such as in how many increments)
   f. Roles (project) and reserved resources
   g. Priority

These aspects refer more specifically to the project organization or the delivery of the solution. Note that during the work so far, all the above results have been examined and analyzed for the solution. In this part, the same results are re-used but the perspective is no longer for the solution but for the specific project that will deliver the solution.

Dependencies state the links to other projects, initiatives or deliverables that the project will have. A stakeholder affected lists all the stakeholders that need to be interacted with during the project such as the stakeholder matrix previously presented. Critical factors list the risk that can affect the project and how they can be mitigated. Gained approval is where all involved departments and managers have given their approval and support for the project. Delivery considers how the project is to be organized in terms of delivering the solution such as in one go or in increments. Roles define the resources and their roles in the project.

Finally, priority determines if the cost, quality or time aspect of the project is the primary priority. Projects have a devil’s triangle consisting of time, cost and quality. For instance, the project cost can be reduced and the speed (time) of the project can be increased to deliver the results earlier. However, such decisions will affect the functionalities that will be included (quality). The priority will affect the decisions taken in the project. If a project has time as main priority and a senior developer gets ill, he or she needs to be replaced. If time is of essence, a senior developer from a consultancy firm can be hired (same quality but higher cost). If cost is the main priority, perhaps a junior developer can step in (longer time but less costly). The priority is set in consultation with the business analyst and the key stakeholders of which the sponsors have more weight in the decision.

---

1 These activities are from Babok v3