Selecting the best candidate for Business Process Redesign

Jesse van Bekkum
March 2010
Abstract

Many enterprises have a lot of business processes and often execute improvement projects to increase efficiency and reduce costs. However, starting an improvement project on all of them is usually not possible due to a lack of time and resources. This project focuses on the very first step of process improvement: doing a quick scan over a large set of processes, and selecting the candidates which are most likely to have room for improvement. These processes can then be looked into in more detail. This project was executed at the department of Information and Process Engineering at ING Investment Management.

In this paper I have developed a method to give a process a score on four different aspects: Cost, Flexibility, Quality and Time. The score for each aspect is determined by answering questions. For this I developed a question model and an accompanying tool that allows presenting the questions to the user where the answers can be used to calculate scores for these four aspects, and suggestions for improvement can be given. The question model is easy to modify in ARIS, a modelling tool used by ING IM. The questions in the question model can be answered by the processes owners, or in a select number of cases, automatically. These questions are related to properties of the process, such as the amount of moving time, the amount of incidents, the required levels of communication, etc. For each question I show a description for the end user, a motivation for asking this question and what information sources can be used to answer this question.

In total I have developed 44 questions, 18 of which can be answered automatically. When a question is answered automatically, external data sources, such as logs and process models are used, to calculate the correct answer. Four automatic questions are supported by a tool implementation.

The answers to these questions is based on that a process model, given in an Event-driven Process Chain (EPC), as well as a set of operation logs. EPC’s were chosen because they are used at ING IM as well as at many other companies. For these four questions a formal formulation is given and an algorithm to calculate the answer.

I describe two use cases where the method was used to analyze a process. Furthermore, I show the usage of the automatic questions on a fictional process with generated logs.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of Contents</td>
<td>3</td>
</tr>
<tr>
<td>Abstract</td>
<td>2</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>5</td>
</tr>
<tr>
<td>1.1 Description of the project</td>
<td>5</td>
</tr>
<tr>
<td>1.2 Context</td>
<td>6</td>
</tr>
<tr>
<td>1.3 Document structure</td>
<td>7</td>
</tr>
<tr>
<td>2 Process improvement at ING IM</td>
<td>8</td>
</tr>
<tr>
<td>2.1 Available information</td>
<td>9</td>
</tr>
<tr>
<td>3 Related work</td>
<td>11</td>
</tr>
<tr>
<td>3.1 Business Process Redesign</td>
<td>11</td>
</tr>
<tr>
<td>3.2 The devils quadrangle</td>
<td>12</td>
</tr>
<tr>
<td>3.3 Incident management</td>
<td>15</td>
</tr>
<tr>
<td>4 Structure of the question model</td>
<td>17</td>
</tr>
<tr>
<td>4.1 My interpretation of the devil’s quadrangle</td>
<td>17</td>
</tr>
<tr>
<td>4.1.1 Quality</td>
<td>17</td>
</tr>
<tr>
<td>4.1.2 Time</td>
<td>18</td>
</tr>
<tr>
<td>4.1.3 Cost</td>
<td>18</td>
</tr>
<tr>
<td>4.1.4 Flexibility</td>
<td>19</td>
</tr>
<tr>
<td>4.2 The syntax of the question model</td>
<td>19</td>
</tr>
<tr>
<td>5 The checklist</td>
<td>28</td>
</tr>
<tr>
<td>5.1.1 Sub processes</td>
<td>31</td>
</tr>
<tr>
<td>5.2 Conclusions</td>
<td>52</td>
</tr>
<tr>
<td>5.3 Scores</td>
<td>58</td>
</tr>
<tr>
<td>5.4 Use cases</td>
<td>59</td>
</tr>
<tr>
<td>6 Partial automation of the checklist</td>
<td>63</td>
</tr>
<tr>
<td>6.1 Definition of EPCs</td>
<td>64</td>
</tr>
<tr>
<td>6.1.1 Syntax</td>
<td>64</td>
</tr>
<tr>
<td>6.1.2 Semantics</td>
<td>65</td>
</tr>
<tr>
<td>6.2 Definitions required to calculate the metrics</td>
<td>68</td>
</tr>
<tr>
<td>6.2.1 Roles &amp; Resources</td>
<td>68</td>
</tr>
<tr>
<td>6.2.2 Logs</td>
<td>68</td>
</tr>
<tr>
<td>6.3 Metrics</td>
<td>68</td>
</tr>
<tr>
<td>6.3.1 Exceptions</td>
<td>69</td>
</tr>
<tr>
<td>6.3.2 Handovers:</td>
<td>70</td>
</tr>
<tr>
<td>6.3.3 Queue time</td>
<td>71</td>
</tr>
<tr>
<td>6.4 The algorithm</td>
<td>73</td>
</tr>
<tr>
<td>6.4.1 Definitions</td>
<td>75</td>
</tr>
</tbody>
</table>
6.4.2 The algorithm: ................................................................. 76
6.4.3 Test of the algorithm ....................................................... 78

7 Conclusion .................................................................................. 80

8 Bibliography .............................................................................. 82

9 Appendix .................................................................................... 83

9.1 Propagate positive tokens ..................................................... 83

9.2 Score figures ........................................................................... 87
1 Introduction

Business process improvement has become an important activity in many of today’s businesses. The goal of process improvement is to improve the performance of a business processes by reducing costs and increasing quality and efficiency. There are several commonly used methods of process improvement, such as six sigma (Tennant, 2001). Common elements in these methods include analysis of the current situation, proposing improvements, implementing these improvements and measuring their effect.

One problem which arises early on is: where to start? Even when the exact methodology has been defined, choosing which process to improve and how to improve it is not easy. Large enterprises have many processes and most organizations cannot focus on all of their processes at the same time. Therefore it is required to select a few processes to start an improvement project. It is best to start with the worst performing processes for these will have the highest probability that significant improvements are possible.

What is required is a method to assess the current performance of a process and find the worst performing process from a larger set of processes. This should be low priced, easy and fast. The processes performing worst in this method are the candidates for more thorough analysis and process redesign.

1.1 Description of the project

Problem definition
The problem addressed in this thesis is the following:

How to select good candidates for process improvement and how to discover what kind of improvements to apply?

The main goal of my thesis is:

Develop a method to quickly determine the relative quality of a process. When applied to a lot of processes, this should show what processes are the best choices to look into in more detail for an improvement project. This method should be fast, easy and cheap to apply. This method should indicate what information about the process is required and, for a number of cases, show how to use the information.

There are a number of constraints:

- The improvements should be achievable by adjusting the process model, not the IT infrastructure. Adjustments to the IT infrastructure are often expensive and require testing, implementation and training.
- Improvement is evolutionary. In general there are two methods to change a process: evolution and revolution. A revolutionary change often involves a complete redesign, which is a lot of work, risky, time consuming and cannot be automated. Evolutionary changes are small steps within a process. In order to keep the method cheap and small scale, only evolutionary improvement suggestions will be given.
Contents of this project
Gathering information about a process is the most important step in analysing a process. The main source of information used is expert knowledge from the process owners. To gather this expert knowledge I used a question model.
The goal of the question model was to:
• Be a powerful method to gain information about a process
• Be easy to understand
• Be easy to update and expand
• Be able to assess a process and give suggestions for improvement.
The assessment is based on four different aspects: cost, quality, time and flexibility.

The implementation of the question model is done in a custom tool. Modifying the question model can be done in existing software. The questions in the question model are based on earlier comparable projects and other work on this subject.
For each question I specify what additional sources of information would be required to answer this question, and whether it is possible to automate answering this question. For some questions I have defined metrics and described an algorithm to answer these questions automatically. The automatic answering of these questions has been implemented in the tool.
The implementation is tested with a fictional process and randomly generated logs.

1.2 Context
This project took place at ING Investment Management (ING IM) in The Hague in the department of Information and Process Engineering (I&PE).

ING Investment Management
ING Investment Management is part of the Dutch ING group that handles investments. Investment management has three main groups of clients: Retail, Wholesale and Insurance. Retail are the investments funds of ING, where regular consumers can invest their money. Wholesale are large, institutional investors such as pension funds. Insurance are the investments from the ING group's own insurance activities, such as life insurances.

I&PE
Information and Process Engineering is the department of ING that supports the investment business by modelling, analysing and improving their processes.
The process team is a part of the I&PE department. Its goal is to monitor and improve the processes of ING IM. In order to make improvement more efficient, they would like a method to assess the performance of a process. The reasons and requests are briefly summarized here:

• At the moment I&PE spends most of its time analysing the processes, and keeping them up-to-date with ever present changes. To use what time is left most efficiently, a method should be available to select which process is suitable for improvement and where to spend effort.
• The method should result in measurement on how healthy the process is and suggestions on what to do.
The method should assist in a strategy to improve processes continuously and should be applied regularly. Furthermore, it should not take too much effort to apply the method. For these reasons the method has to be supported by a tool.

1.3 Document structure

In this thesis I first summarize some related work on this subject, and describe the environment in which this project was done. In the next part I present a large set of multiple choice questions which I use to apply the theory in practice. Each question is a multiple choice question, and includes a description for the process owner, rationale and where information about this question can be found. From the answers to these questions scores and conclusions can be generated. Use cases of the change management process and the settlement process are presented.

In many cases it is more accurate and more objective to calculate the answers to certain questions using data about the process instead of asking the process owner. The next part explains an analytical approach to analyse logs of a process, and calculate certain metrics. These calculations can be combined with the normal questionnaire.

In the conclusion I summarize the problems encountered and indicate where further research is necessary.
2 Process improvement at ING IM

This project was done at the Investment Management part of the ING group. In this chapter ING IM and their process improvement efforts are described. Also included are descriptions of different available data sources, and the usefulness of the data as input for this project.

ING Investment Management

ING Investment Management manages the investments for the ING Group. It has three main groups of clients: retail, wholesale and insurance. The complete value chain is divided into six parts: Marketing and Sales, Client Servicing, Trading, Transaction processing and Accounting and Reporting.

The first contact a client has with ING IM is with Marketing and Sales. They market the services of ING IM, and find customers who want to have their money invested by ING IM. In client servicing a mandate is created. A mandate is an agreement on what kind of items to invest in with respect to risks, geography and other preferences of the customer.

With the mandate an investment manager can start to invest. He decides what stocks to buy, and makes sure the stocks he buys are in confirmation with the mandate of the customer. Once he has decided to buy a stock, a trader actually buys the stocks from the stock exchange. Then, within a couple of days, the trade needs to be settled. This is the actual payment for the stocks. Afterwards Accounting and Reporting reports to the customer the result of the investments.

The secondary processes are the support departments. They support the primary processes in various ways. This includes I&PE, whose job it is, among other things, to document and improve the processes. All together there are about 120 distinct processes.
The method of this project will apply to the primary processes. This is because the primary processes are more “factory like” processes, with a higher volume of cases and clearer function descriptions. Moreover the primary processes handle much larger amounts of money and are more important than the secondary processes.

**The Scorecard**
Before my assignment I&PE developed a so called scorecard for this exact problem. It focused on five points: time, cost, output quality, process quality and risks. For each of these categories questions had to be answered, such as the number of FTE involved in the process, various key performance indicators, handovers and execution time. All questions had to be answered manually and there were no plans to for automation. These answers were to be compared with the values the process owners thought were desired in such a process.

This scorecard has never been used in practice. The main problems were the time process owners could spend on it and the difficulty of obtaining the data. Some parts were used as inspiration for this method.

**2.1 Available information**
The most important source of information is expert knowledge of the people involved in the process. However, this information might be subjective and inaccurate. It is preferable to use objective data from the processes to base a score on. One of the first things I did was investigate which data could be obtained.

**Workflow models**
ING IM has three defined levels of process abstraction. These are the processes, procedures and work instructions. Processes are the highest level and regard information flows and activities between different departments. Procedures are a level beneath, for departments internally. Work instructions are the lowest level, with clear descriptions and screenshots. The process team usually works on the process level and incidentally on procedure level. The rest of the procedures and the work instructions are created and maintained at department level. It is not feasible for the process team to know everything on that level of detail and keep it up-to-date.

The processes are modelled in Event-driven Process Chains, in the modelling program ARIS. One problem with the process models was that they are intended to be read by humans, not by computer programs, which means a lot of information is modelled informally. For example, the four eyes principle is only described in text, not as a formal parameter. This will make it harder to use the process models in calculations. The process models have several uses, such as describing the processes to people unfamiliar with them, showing how the procedures and work instructions relate to each other and showing regulators that certain controls and regulations have been. The process models are not used for any analysis regarding performance.

**KPI's**
For every part of ING IM certain Key Performance Indicators (KPI) are defined and collected on a quarterly basis. A KPI measures some specific performance issue, like the amount of reports that were delivered on time or the number of cases that went straight through the process without needing any correction. All KPI's can be linked to certain processes, and can be used to see what processes are not running perfectly.
Combining this with the data from the workflows might show if some bad performing KPI's have an influence on other KPI's.

**Logs**
The analysis of logs is a very powerful method to gain information about a process. They can be analyzed automatically, and several indicators like the amount of rework and the average queue time can be calculated. For thorough analysis the logs have to be sufficiently detailed. If only the time a case enters the process and the time it leaves the process are recorded, it is impossible to calculate anything about the behaviour inside the process. If for each case it is recorded when it entered and left each activity, more calculations can be made.

It is possible to record detailed logs manually, but this will be too expensive in almost all situations. Therefore useful logs will only be available in cases where the process is partially automated (people who are supported by a business process management system, for example) or fully automated. Changes to a fully automated process will often require modifications in the IT systems, and as described in the first chapter, changes in IT systems are considered beyond the scope of this project. Therefore partially automated logs are the only logs that are relevant to the project.

At ING IM, the logs available are all from completely automated systems. Unfortunately there are no logs of partially automated systems.

**Incident reporting**
ING IM has an incident reporting system. An incident is an event that costs money, has a large risk to cost money or has an impact to reputation, safety or health. This can be things like wrongly executed transactions, costs due to failed deadlines, etc. Incidents like these are to be reported in an incident management system, IR-scan. Data from this system could be used to determine quality issues within certain processes.

A drawback of this system is that all incidents are entered manually. This means that the total number of incidents in the system is not very high and that automatic analysis is not required. At the moment incident analysis is done by humans, and any automation will not yield any startling conclusions. The low number of incidents will mean that outliers have a very large influence on the results, making automatic analysis unreliable.

**Auditing**
As is required for financial institutions like ING there are several levels of auditing. These levels are the Dutch supervisor of the financial markets (AFM), the accountancy firm Ernst & Young, the Corporate Accounting Services (CAS) and the management teams. The last two are internal to ING IM and report their findings into AO-scan, a computer system for this purpose. The internal auditors choose several focus points each year and do random checks. Each issue they find is assigned to a responsible person. Later this person is to report the improvements on the issue. A disadvantage is that an audit is only specified at a high level of abstraction. The lowest level usually specified is the value chain, which is still far above the level of individual processes. Because audit items are entered manually the total number of items is not extremely high, making automation unnecessary.
3 Related work

Business Process Improvement, also called Business Process Reengineering and Business Process Redesign, aims at improving the performance of existing business processes. Whatever strategy is used, the following elements occur in all methods: Planning, executing the plans and measuring the effects. Part of the planning stage is selecting what process to focus on. The relevant part for this project is selecting a process.

In general there are two approaches to business process redesign: evolutionary and revolutionary. The revolutionary approach is a clean-sheet design: throw everything away and start anew. Evolutionary redesign is to take small steps, and improve the process incrementally. The advantages of revolutionary redesign are that you do not start with any inheritance from the old design, which can result in larger improvements. The disadvantage however, is that revolutionary redesign projects are larger and often cross process boundaries, making them more costly and riskier. I will focus only on evolutionary redesign.

3.1 Business Process Redesign

One of the first people who did research in the redesign of processes in administrative environments was Thomas Davenport (Davenport, 1993). However, his approach is quite abstract. Like most methods he first focuses on modelling the current situation and on implementing a new design. The method Davenport describes consists of the following five steps:

- Enumerate major processes
- Determine process boundaries
- Assess strategic relevance of each process
- Render high-level judgments of the “health” of each process
- Qualify the culture and politics of each process.

Davenport devotes a lot of time on the first two steps and their execution, but glances more abstractly over the other three steps. He mentions that indicators of an unhealthy process are the amount of queue time compared to the amount of service time, the number of handovers, whether a process has a clear owner and a clear customer.

Another important book on business process reengineering is the book “Reengineering the corporation” (Hammer & Champy, 1993). They identify three key indicators in choosing the process to reengineer: dysfunction, importance and feasibility. Dysfunction is whether a process performs well at the moment or not, importance is whether a process is important to the customers and feasibility is whether process improvement is realistic. There are many symptoms of dysfunction, including:

- Extensive information exchange, data redundancy and rekeying
- Inventory, buffers and other assets
- High ratio of checking and control to value-adding
- Rework and iteration
- Complexity, exceptions and special cases

Best Practices

Hajo Reijers has done a lot of research on redesigning a process. In his book (Reijers, Design and Control of Workflow Processes, 2002) he identifies a total of 29 different redesign techniques, which he calls the “Best Practices”. These practices are derived from other literature in the field. A best practice is an atomic change in a process which improves it.
For example, the best practice of *task elimination* means that a task which is not necessary, like a control that is done twice, is removed. This will of course speed up the process, but might reduce quality. *Parallelism* is having tasks that are done sequentially done at the same time, in parallel. This might speed up the process, but can also increase start-up time because of the higher probability that a task is executed by two different resources.

These best practices form a comprehensive list, intended to cover almost any process change using one or more best practices. The appealing property of the best practices is that they give a suggestion on what the direct and indirect effects of applying a practice will be. When, for example, quality is an important issue it is easy to see what practices can be looked at first and which ones will have an effect in the opposite direction.

### 3.2 The devils quadrangle

Redesigning a process can have different results. The execution may be faster, fewer people may be required or higher quality may be delivered. There are several methods to measure process performance. A useful and often used method is the devils quadrangle (Brand & Kolk, 1995).

The “devils quadrangle” is a quadrangle with four dimensions: *time, cost, quality* and *flexibility*. I have chosen the devils quadrangle because it covers the most important aspects correctly and it is used in a lot of other papers I have used. The main difference between this model and other models is the use of the flexibility aspect. Quality, cost and time are usually present, often combined with some variant of risk. The flexibility aspect covers risks and also includes some other interesting properties, such as the ability of the process to cope with different volumes.

![Figure 2: The devils quadrangle](image)

The further away from the origin a factor is, the better it is. That means that the devils quadrangle moves away from the origin with an *increase* in flexibility and quality, and with a *decrease* in cost and time.

The problem with the devil’s quadrangle is that when improving one axis, it often has a deteriorating effect on some other axis. For example when adding resources (hiring more people), jobs will be finished faster, at a higher cost. That means that the time axis will
improve, but the cost axis will decline. If a control task somewhere is removed, the case will be processed faster and resources have to work less, but it is more probable that errors will remain unnoticed. This results in an increase in the time and cost axis, but a decrease in the quality axis. Hence the name devil’s quadrangle, for there is often no perfect solution.

The four dimensions quantify the different aspects of a process. The definition of Kolk and vd Brand is rather limited, but a good list of sub metrics can be found in (Jansen-Vullers, Kleingeld, & Loosschilder, 2007), which I have repeated here. I have based my interpretation of the devils quadrangle in chapter 4.1 on this list.

**Time:**
- Service time: the time that resources spend on actually handling the case
- Queue time: the time a case spends waiting in a queue
- Wait time: all other delays for a case, e.g. the time a case has to wait in
- A parallel branch for completion of all other branches, in order to be able synchronize
- Move time: the time it takes to move a case between tasks
- Setup time: the time it takes to setup a task for a case, for example the time to get acquainted with the case

**Cost**
- Running costs: costs for executing the workflow
  - Labour costs: costs of the workforce (salary * worked hours)
  - Machinery costs: an investment in machinery per new automated task
  - Training costs: costs for training employees (based on the number of employees and the number of new tasks)
- Inventory costs: costs of keeping records and products (e.g. file cabinets)
- Transport costs: costs for moving (intermediate) products and sharing information (distance * price per km)
- Administrative costs: costs for keeping the entire workflow intact
- Resource utilization: the ratio of in use time and available time of resources

**Quality**
Quality can be divided in internal and external quality. External quality is the quality from the perspective of the customer. Internal quality is the quality of a process from an operator’s perspective.

**External quality**

*Quality of the output*
- Performance: the degree to which a product's primary operating characteristics meet customer's requirements
- Conformance: the degree to which a product's design and operating characteristics meet established standards
- Serviceability: the speed, courtesy, competence and ease of correcting mistakes

*Quality of the process*
• Information on application status: the degree to which information on the application status is provided
• Bureaucratic language simplification: the clearness in the presentation
• Information availability: the time required to get updated on the status

Internal quality
• Skill variety: the degree to which a job requires a variety of different activities in carrying out the work, which involves the use of a number of different skills and talents of a person (number of different tasks and case types).
• Task identity: the degree to which the job requires completion of a whole and identifiable piece of work; that is, doing a job from beginning to end with a visible outcome (ratio of number of executed tasks and total number of tasks per workflow).
• Task significance: the degree to which the job has a substantial impact on the lives or work of other people, either in the immediate organization or in the external environment.
• Autonomy: the degree to which the job provides substantial freedom, independence, and discretion to the individual in scheduling the work and in determining the procedures to be used in carrying it out the work (ratio of number of authorized decisions and total number of decisions).
• Feedback: the degree to which carrying out the work activities required by the job results in the individual obtaining direct and clear information about the effectiveness of his or her performance.
• Co-worker relations: the quality of the relations between an employee and his/her co-workers.

Flexibility
• Mix flexibility: the ability to process different kinds of cases
  o for resources: number of case types a resource can handle
  o for tasks: number of case types a task can handle
  o for the workflow: number of case types that can be handled
• Labour flexibility: the ability to perform different tasks
  o for resources: number of executable tasks
  o for the workflow: available resources per task per case
• Routing flexibility: the ability to process a case using multiple routes (number of different sequences in the workflow)
• Volume flexibility: the ability to handle changing volumes of input (available time per employee)
• Process modification flexibility: the ability to modify the process (number of sub flows in the workflow, complexity, number of outsourced tasks, etc.)
3.3 Incident management

Part of the quality aspects are errors and incidents. Errors and incidents are usually caused by mistakes made by operators. Some errors are corrected, others are very small and go unnoticed, and some cause real problems and cost money. For further analysis errors and incidents I have used theory from safety management. Safety management is the analysis of industrial and transportation accidents, for example fires in chemical plants or airplane crashes. These theories can also be applied to administrative environments.

![Diagram of incident causation](Van der Schaaf)

On the left hand side in Figure 3: Incident causation (Van der Schaaf) are the three types of errors that can cause an incident. The first is technical failure. These are errors caused by a broken or inadequate designed machine or system. Then there are human failures. These are mistakes made by people. Finally there are the organizational failures. These are errors caused by the organization, for example inadequate procedures or insufficient training.

The technical and organizational errors are in the end always caused by humans (mistakes in designing machinery or setting up procedures) but they are classified as technical or organizational mistakes.

If something goes wrong, this creates a *dangerous situation*. This does not have to be a problem, since there are defences against errors. These are necessary, because in every organization mistakes are made, and especially human mistakes are never prevented completely. Defences can be of technical nature or of organizational nature. Examples of a technological defence are warning lights in a car, checks in software packages fuses in a fuse box, etc. Examples of organizational defences are checklists, people whose task it is to perform checks, etc.

The planned defences prevent most errors, but sometimes an error slips through these defences. Then there is always the possibility of unplanned recovery. This is when someone sees the error even though it’s not the person’s task to look for errors. For example a patient who discovers the pills he got are different than his usual ones, a passenger who alerts the flight crew about a defect, etc.
This kind of recovery is usually based on alertness, experience and sheer luck and cannot be relied on. If this happens it is called a near miss, otherwise it is an actual incident.
4 Structure of the question model

I have developed a tool to support the process improvement. The tool has two functions:

Scoring
The processes are scored on four factors. Four of those are the factors in the devil’s quadrangle. The scores are quite elementary: a positive aspect for a certain factor gives +1 point, a negative aspect -1 point. It is possible to give certain aspects a higher weight. A text can be provided to explain certain scores.

Conclusions/Suggestions
Based on the answers suggestions are given. This is often the application of a certain best practice. When certain aspects are selected some best practices are suggested. More detailed explanation is provided.

In this chapter a checklist is described which is used in the tool. The goal of the checklist is to systematically look into all the factors of the devil’s quadrangle, in order to find out where risks and problems of the process are, and what strategies to apply. In the tool the checklist is presented as a list of multiple choice questions. Which questions are asked depends on the answers given.

One of the goals of the tool is to make modifying and maintaining the questionnaire easy. For that reason the questionnaire is modelled in ARIS with EPC elements, although the model is not an EPC. The result is a powerful model which can be edited with tools regularly used at ING and without much specialist knowledge.

4.1 My interpretation of the devil’s quadrangle

The question model scores a process on four different aspects. However, not everything in Jansen-Vullers definition of the devil’s quadrangle is usable for the situation in my project. For example, the cost metrics of transport, inventory and machine running cost are specific to processes with physical goods. Move times are effectively zero, because all that moves is information, which is transported electronically and instantly. In this section I describe what parts of the devil’s quadrangle I use in the scoring model.

4.1.1 Quality

In the definition of Kolk and Vd Brand there are two kinds of quality: internal and external quality. Only the external quality aspect is used in this project.

Internal quality, quality from the perspective of the operator, is very difficult to measure. Furthermore, the internal quality has no direct effect on the performance of a process. While more varied and challenging work may increase quality on the long term (because people stay longer, allowing them to gain more experience), it has no direct effect on performance. Internal quality is considered to be outside the scope of the project.

External quality is the quality from the perspective of the customers. The most important factor for customers is the return on their investments. However, anything related to the quality of the decisions of what funds to invest in, and what funds not, is outside the scope of the project. Used in this project are:
• Output quality. This is anything that contributes to the quality of the errorless output of a case. The amount of cases that go through a process without needing any corrections, the number of incidents and satisfaction of (internal) customers.
• Process quality. Anything related to the process itself. The number of cases that take a path through

4.1.2 **Time**
The *time* axis is the time necessary for the execution of a case. This consists of the:

• Service time. The time actual work is done on a case by a resource. Service time is difficult to improve. Only technological advances or training can increase the speed people are working.
• Lead time. This is the total time a case is in the process, from the initial contact with the customer until the delivering of the product and any associated administration. The lead time has two important factors, the average time and the deviation from average. Deviation is an important factor, because stakeholders are often more interested in the number of items that are late than the actual average lead time. The minimal lead time is the service time, but this is extremely rare and usually means that there are too many resources available.
• Queue time. The queue time is the amount of time a case is waiting in a queue for a resource to become available. In practice this means the case is on someone’s to-do list, or it is in his mailbox. In a lot of administrative processes, the time a case spends in queues is surprisingly high compared to the time someone is actually working on it. The queue time can be up to 99 percent of the total lead time.
• Wait time. This is the time a case is waiting, while no resources are working on it. An example is if a decision about all cases is always made during the Wednesday morning meeting. If a case arrives at Thursday morning, it will wait for 6 days.

4.1.3 **Cost**
*Cost* is the direct cost of the resources of a process. In most administrative settings this is the salary and additional costs of employees working on a process. In an industrial setting this also includes the cost of machinery. It must be clear that only direct running costs are used here. In the end, everything has some cost. People who correct mistakes cost money, people working overtime cost money, etc. However, these are quality and time issues. If these other costs were included, the cost category would become too broad and too overwhelming to the other factors.

The cost factor can be divided in several sub metrics:

• Running cost. These are the costs of the wages paid for useful, necessary work. These costs are generally unavoidable, since work which has to be done just has to be done. Decreases in running costs are usually due to technological improvements, like automation of certain tasks. It is not possible to decrease these costs on a process improvement level.
• Setup time cost. Setup time is time required to start-up working on a case. This can be booting a computer, configuring a machine or reading into the specifics of a case. The difference with service time is that this time (and cost) can be spread out if multiple similar tasks are done in succession, thus decreasing the amount of setup-time per case.
• Idle time cost. This is time that insufficient work is available for workers and workers are not working at their top level of efficiency.
4.1.4 **Flexibility**

Flexibility is the possibility of the process to adapt to changes and to handle unusual situations. Flexibility itself is difficult to measure, but if there is a lack of flexibility it will usually result in lack of time.

**Volume flexibility (or demand variation)**

The number of cases a process has to handle is not always constant. This is called demand variation. Every business has to deal with a variation in the demand of its products or services. A supermarket cannot predict the exact amount of beer they are going to sell on a single day, a doctor cannot predict when patients are going to visit, a fireman can’t predict when there is going to be a fire. In investment management, variation in demand is usually determined by the market: some days just a few trades happen, on other days there is a lot of trading. And the market affects the whole organization: If the traders trade a lot, settlements has to settle a lot, reconciliation reconciles a lot, etc. Demand variation is generally a bad thing, but it is unavoidable. Some companies try to reduce the variation in demand, for example the Dutch railways. By providing a discount card they try to encourage people to use the train in the low traffic hours.

**Mix flexibility**

A process can get different kind of cases. They may differ in terms of difficulty, time to process, resources required, etc. If the proportions of the different kind of cases can change this can have influence on the process. This is called mix flexibility.

For example, the following process:

![Figure 4: Mix flexibility example](image)

This process can have two kinds of cases, cases where the European law applies and cases where American law applies. If, for some reason, there are a lot of European cases and just a few American cases, the American expert will have nothing to do while the European expert is overworked.

### 4.2 The syntax of the question model

The question model is modelled in ARIS. While EPC elements are used, the syntax itself is different from standard EPC’s.

**The syntax**

The following elements are used:

<table>
<thead>
<tr>
<th>ARIS</th>
<th>Questionnaire model</th>
<th>Picture</th>
</tr>
</thead>
</table>
The model is a directed, acyclic graph. The graph is like a tree, but it is possible for branches to merge. The user starts in the root node. The graph contains questions, and the user traverses the tree according to the answers given to questions. Also included in the graph are “sub conclusions”. Each sub conclusion acts as a flag that records the path taken through the graph.

The conclusion elements model conclusions, suggestions for improvement for the user. The score elements model a score for the process. They do not occur in the main graph, but in a set of smaller, disjoint graphs.

The following rules have to apply to the model:

- The questionnaire contains exactly one event with the name “STARTSTART”. This is the start of the model.
- Each sub model contains exactly one event with the name “START”. This is the start of that sub model.
- A “START” event or a “STARTSTART” event is followed by a question (if one question follows that event) or an and-node (if multiple questions follow that event).
- Each question is followed by an or-node, or a xor-node.
- or means that the question is of the “Check all that apply” type. Zero or more answers are correct. In the program it is presented as a list of checkboxes
- xor means that the question is of the “Check only one answer” type. Only one answer. In the program it is presented as a list of radio buttons

<table>
<thead>
<tr>
<th>Function</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>Answer</td>
</tr>
<tr>
<td>Automatic function</td>
<td>Sub conclusion</td>
</tr>
<tr>
<td>Control function</td>
<td>Conclusion</td>
</tr>
<tr>
<td>Control function</td>
<td>Score element</td>
</tr>
<tr>
<td>Sub process</td>
<td>Sub model</td>
</tr>
<tr>
<td>Operating unit</td>
<td>Automatic question indicator</td>
</tr>
<tr>
<td>Position</td>
<td>Automatic answer indicator</td>
</tr>
<tr>
<td>XOR</td>
<td>XOR</td>
</tr>
<tr>
<td>OR</td>
<td>OR</td>
</tr>
<tr>
<td>AND</td>
<td>AND</td>
</tr>
</tbody>
</table>

The model is a directed, acyclic graph. The graph is like a tree, but it is possible for branches to merge. The user starts in the root node. The graph contains questions, and the user traverses the tree according to the answers given to questions. Also included in the graph are “sub conclusions”. Each sub conclusion acts as a flag that records the path taken through the graph.

The conclusion elements model conclusions, suggestions for improvement for the user. The score elements model a score for the process. They do not occur in the main graph, but in a set of smaller, disjoint graphs.

The following rules have to apply to the model:

- The questionnaire contains exactly one event with the name “STARTSTART”. This is the start of the model.
- Each sub model contains exactly one event with the name “START”. This is the start of that sub model.
- A “START” event or a “STARTSTART” event is followed by a question (if one question follows that event) or an and-node (if multiple questions follow that event).
- Each question is followed by an or-node, or a xor-node.
- or means that the question is of the “Check all that apply” type. Zero or more answers are correct. In the program it is presented as a list of checkboxes
- xor means that the question is of the “Check only one answer” type. Only one answer. In the program it is presented as a list of radio buttons

20
• The xor- or or-nodes are followed by at least two events. These are the answers to the question.
• The answers are followed by either:
  o A light blue function. This is a sub conclusion.
  o An and-node ($\land$).
  o Another question
  o Nothing
• The and-node will lead to at least two new questions. This allows to add multiple questions after a certain answer.
• The sub conclusion is followed by the same items as the answer.
• Automatic question indicators have to be attached to the question they apply to.
• Automatic answers indicators have to be attached to sub conclusions behind the answers they apply to.

Automatic questions
Some questions can be answered automatically as well as manually. If an automatic question indicator is connected to a question, this question can be answered automatically. An automatic question always has only one answer, and is followed by a xor. The xor is followed by answer elements, and these answer elements are followed by sub conclusions. The automatic answer indicators have to be attached to the sub conclusions, as in Figure 5. Not all answers need an automatic indicator (like D). But these answers will not be used in an automatic answer and can only be selected manually. It is important that the text in the automatic indicators (the yellow elements) is fixed, since it is used in the program itself to interpret the model. The text of the question and answers can be changed for the user.
Figure 5: An automatic question

Scores
Assume there is a set $S$ for all sub conclusions that were flagged during the calculation of the model.

For each score metric there is a tree with connectors in the nodes, sub conclusions (blue functions) in the leaves and conclusions (red functions) in the root and between connectors. For the score for a certain metric A:

- The root node is “Score: A”
- Before that node is an **and** connector
- Before the connector two conclusions with “Score: Good A” and “Score: Bad A”. The good conclusion collects all factors that are positive for A; the bad conclusion collects all factors that are bad for A.
- Before those two conclusions two or nodes
- Before those two or nodes the leaves: Sub conclusions leading to this score.
Figure 6: Score definition

Figure 6: Score definition presents a graphical example. Factor 3 is positive for A, 1 and 2 are negative for A. Each sub conclusion preceding a “good” score element is counted as +1, each sub conclusion preceding a “bad” element is counted as -1.

There is no requirement for the good factors and bad factors to be balanced. For example, it is possible that there are only 2 good factors, and 9 bad factors for a particular metric. A score is shown as a traffic light. A metric has a green score if the score is in the best 33% of the total possible range (a score between -1 and 2 in the example). Red is in the lowest 33% (a score of -9 to -5 in the example) and yellow in between (-4 to -2). In some cases it is actually possible to have a green score with a negative result.

Conclusions

For each conclusion there is a simple tree with the sub conclusions leading to that conclusion.

- A conclusion is preceded by a connector or a sub conclusion.
- The connector is preceded by other connectors or by sub conclusions.
- Whether a certain conclusion is true is calculated using the elements in $S$ and the normal logical rules of the connectors.

Figure 7: Conclusion definition

Conclusion 1 is considered true if $Factor\ 1$ is in $S$. Conclusion 2 is considered true if $Factor\ 2$ and $Factor\ 3$ are in $S$. Instead of a and-connector a or- or xor-connector could be
used. More elaborate trees with multiple connectors are also permitted, although they are not used in this thesis.

**Descriptions**
In ARIS each element can have a description. This is the only parameter used in the model.
- For questions, the description is shown below the question, to give additional information about how the question should be interpreted.
- For conclusions, the description is shown at the end, giving more information on the conclusion.
- For the score, the description is shown next to the score when clicked on at the end, showing what the score is based on.
- For answers, the description is shown as a tooltip over the answer in the list.

**Example**
This is an example of a model of a simple coffee machine:

![Coffee machine diagram](image)

**Figure 8: Coffee machine**

The green rounded rectangles are the questions. A question is followed by a logical symbol. These are the **AND** ($\land$), the **OR** ($\lor$) and the **XOR** ($\times$), depending on the number of answers that can be given. The pink hexagons are the answers. The light blue rounded rectangles are so called “conclusion points”. When one of these points is passed, it is “flagged” as such. The process starts with a hexagon with the text “START”. Note that the OR has the
meaning of all that apply. That means that none is a valid answer to an OR node, in contrary to regular logic.

The questions are all asked in the tool. See Figure 9.

![Figure 9: Screenshot of a question](image)

At the end, the sub conclusions can be used to derive end conclusions. In Figure 10: Conclusions, one conclusion is that if a user wants a coffee with sugar, he should press button A. For a tea with milk he should press button B.

![Figure 10: Conclusions](image)
Figure 11 displays the result of selecting a coffee with sugar. All texts can be modified from the model.

Figure 12 is the score diagram for a health metric. It is assumed that tea is good for one’s health and sugar and caffeine are bad. A tea would score a green score (+1), a decaf coffee would score orange (0) and a regular coffee with sugar would have a red score (-2). In Figure 13 is the score of a tea with sugar. Tea is positive, sugar is negative and the result is neutral. On the right is a more thorough explanation for this metric.
In the following section the model is described. Each question is listed, with the question (Q), the list of answers (A), the description in the model (D), the reason the question is asked (R) and what data is required to answer this question (I). It is also indicated whether this question could be automated. If the information needed is knowledge from a domain expert, it is not possible to automate. This is not explicitly indicated.
5 The checklist

The checklist scores a process on four points: quality, flexibility, time and cost. Most of the processes contain sub processes. There are some sub processes that occur multiple times in the questionnaire, for example because they affect both time and cost. The tool guarantees that these questions are not asked twice.

![Diagram of the question tree](image)

Figure 14: The root of the question tree

Figure 14: The root of the question tree is the root of the three. It immediately branches into the four aspects that are used. It is possible to select only some aspects, but the different aspects do affect each other. It gives the most complete results if all branches are used. Each of the four aspects branches in a number of sub trees. The branches are shown in Figure 15: Cost, Figure 16: Flexibility, Figure 17: Quality and Figure 18: Time.
Figure 15: Cost

- Exceptions
- Communication
- Start-up time
- Utilization time

Figure 16: Flexibility

- Demand variation
- Workflow
- Case difference
Figure 17: Quality

- Communication
- Demand variation

Figure 18: Time

- Queue
- Exceptions
- Start-up time
5.1.1 Sub processes

5.1.1.1 Case difference

Q1: Are there clear differences between cases?

A: Several case types can be defined, in processing time or difficulty for example; All cases are very standard

D: Often, several categories of cases can be determined. For example: some cases are a lot of work, others are not. Some cases that require a specialized person, some cases that are significantly more difficult than others, etc. Can such distinctions be made here?

R: Not every case is equal. Cases may have different aspects, such as the resources required, the time required, the path through the process or the difficulty. It is of course important that this kind of difference can be detected before. The main risk with different tasks is that if for some reason the proportions of the different tasks would change, that may have all kind of performance effects on the process, such as increasing the load on certain resources, increasing lead time, etc.

The other reason to know this kind of differences is the option of implementing a triage.

I: Logs, detailed information of cases. In some situations it is possible to link certain properties of a case to service or lead time using machine learning algorithms.

Q2: What kind of difference is there?
A: Large difference in process time; Difference in the roles required; Difference in difficulty; Other
D: Select the kind of difference between cases.
R: Try to pinpoint the type of difference. More options may be required in the future. There can also be a difference in path, but this usually is a difference in time or difficulty.
I: Identical to previous question

Q3: Is it possible that the proportions of the case types change?
A: Yes; Unlikely; No
D: If there are e.g. two kind of cases (A and B), and the proportions are that 30% of the cases is type A and the rest is type B, is it possible that those proportions vary or change suddenly? For example, the next day the proportions are A: 60% and B: 40%.
I: Logs: Once several different classes of cases have been identified, it is easy to plot demand over time.
5.1.1.2 Communication

Q4: How many communication steps are there within the process?
A: 2; 3,4; More than 4
D: A communication step is a formal, modelled point where there is contact with a party outside of the process. This can be another ING IM department, or an external party. In cases where there are multiple paths available with a different number of communication points, assume the path with the most number of communication steps. This includes only manual communication; automatic synchronization with outside systems can be ignored.
R: Steps with outside communications are interesting for several reasons. They might introduce delays waiting for reaction from the other party, and they make it more complex to change the process. The minimum number of times is two (getting the information for a case and delivering the finished case to the customer).
I: Expert knowledge. An expert should be able to tell for each function whether it is a communication function. This could also be parameterized in the process model.

Figure 20: Communication
Q5: What’s the amount of informal communication?
A: A lot; Every now and then; Informal communication is rarely required; Unknown
D: Besides the formal communication that is written down in the models, there is also informal communication. This is communication required for execution, but not formalized or modelled. Usually it is in the form of sending an additional email requiring extra information, calling someone to ask a question, walking by the office to discuss a few details about a case, etc.
R: Hammer and Champy: “Good organizational boundaries should be relatively opaque”. If too much communication is needed outside of the official channels, it could either mean that the official channels are insufficiently defined, or that the two organizations ought not to be different organizations in the first place.
I: Expert knowledge. This information is difficult to obtain. Informal actions are by definition not logged.

Q6: What are common reasons for informal communication?
A: Case information is incorrect; Case information is incomplete, Other
D: What are the reasons informal communication is used?
R: When informal communication is used, it is obvious that that information should have been included in the original case. That would have saved time.
I: Expert knowledge

Q7: How well is the input for your product defined?
A: The formats as well as the preconditions are written down; A format has been agreed upon on paper; The format of new cases is not formal
D: In every process there are incoming cases, and those cases are always in some kind of format. A format specifies exactly what information is included and in which form. Also important are preconditions. Preconditions specify the exact conditions the data included is compliant with, and what checks have been done. For example, a case is already checked on compliance with law x, or the information in the case has been checked on correctness with source y. This kind of specifications can avoid double work and decrease errors.
R: The input for a process should be defined clear and unambiguous.
I: Expert knowledge.

Q8: How well is the output for your product defined?
A: The formats as well as the postconditions are written down; A format has been agreed upon on paper; The format of new cases is not formal
D: Good definitions for output are as important as those for input. Formal specifications make it easier to measure the quality of your work, allow your customer to know what they can trust on and what they have to do themselves, etc.
R: The output for a process should be defined clear and unambiguous.
I: Expert knowledge.
Q9: Does the process have contact with the same party multiple times?
A: Yes; No
D: Is there a path possible in the process which communicates twice with the same party? This does not include receiving a case from a customer, and returning a result. For example, when a customer gives a case, and later on in the process, in some cases, the customer is contacted again for additional information.
R: The main reason to ask this is to explore the possibilities of the contact reduction best practice. Communication costs time, induces errors and reduces flexibility.
I: Expert knowledge

Q10: Are any checks done on the data delivered by another party?
A: Yes; No
D: Are there any tasks checking the completeness or correctness of data received from outside the process?
R: Checks on data do not add value to the case, so they can be a candidate for removal, especially if the same check is done on some other place.
I: A computer program could look for certain keywords in process model to detect possible control functions, and let a user verify that.

Q11: Is that party internal or external to ING IM?
A: Internal; External
D: Is the other party within the organization or not?
R: Internal data should not be checked in the beginning of the process, but instead at the end of the process that generates it, and guaranteed to be correct. That makes sure that the correction does not need to go through process boundaries and makes it clear who’s responsible for correctness. For external data it may be required to check correctness, but it should be considered whether elimination is possible?
I: Expert knowledge

5.1.1.3 Customers

![Process Model Diagram]

Q12: Do you know who your customers are?
A: Yes; No
D: A customer is a party who uses the service you deliver. They deliver a request for the service in some form, and later you deliver a result based on that request. A customer should be a clear defined party, and you should know where to send a bill if that would be required.
R: Having identified your customers (even internal ones) is an important measure for process quality (Davenport, 1993). It allows you to determine clear boundaries between processes and makes it easier to measure product quality (customer satisfaction). If your customers are clear, it means for them that their supplier is clear and that they have a clear point to contact if there’s something wrong.

Complement to knowing the customer is knowing the product itself. To sell a product it is important to have exact specifications of the product.
I: Process models or expert knowledge. Each process model should be connected to previous or subsequent models, or the customer should be defined in certain functions.

Q13: Are your customers satisfied with your performance?
A: Very satisfied; Just a bit satisfied; Not at all satisfied
D: How is the quality of your work regarded by your customers?
R: Many theories and methodologies, such as Six Sigma focus on the customer, and adding value for the customer. As such customer satisfaction is an important measure.
I: KPI, from the customer

Q14: If a case is late or incorrect, do you know who to contact about that?
A: For every case we know a responsible person to contact who knows about that case; We have a standard contact who will redirect us; We just call the department, they’ll get back to us why a case is late/incorrect
D: Do you know for each case that enters your process who is responsible and who should be noticed if a case is late?
R: One of the positive effects of a customer structure is the clear responsibility for a case. If somebody feels responsible, and somebody actually notices when a case is late, that’s a feature of a healthy process. (Davenport, 1993). Setting a case manager is also seen as a positive measure (Reijers, Design and Control of Workflow Processes, 2002). Making someone responsible will increase quality because he will feel responsible, and make solving any problems easier.
I: Expert knowledge. It can be found in the procedures.

Q15: What are the main complaints of your customers?
A: Items are late; Average processing time is too high; Too many errors in product; Other
D: For what reasons are your customers not satisfied.
R: Gives a measure what is actually important to improve. Many methodologies consider the customer as the main input for a process.
I: KPI’s, Expert knowledge

Figure 23: Suppliers

Q16: Do you know who you require services from?
A: Yes; No
D: Have you clearly defined your suppliers?
R: Since you are a customer to your suppliers this is equally important as your customers.
I: Expert knowledge.

Q17: Are you satisfied with your supplier’s performance?
A: Very satisfied; just a bit satisfied; Not at all satisfied
D: How do you regard the performance of your suppliers? Would you keep using their services if you had the option to switch to another provider?
R: Customer satisfaction is planned to introduce as a measure/KPI
I: Expert knowledge, KPI’s

Q18: What's wrong with your supplier’s performance?
A: Suppliers make a lot of errors; their work takes a lot of time; they are often later than promised; Other
D: What are your main complaints about your suppliers?
R: If time suppliers are unreliable to deliver on time or correctly, the reliability of this process will also suffer.
I: Expert knowledge

5.1.1.4 Demand variation

Figure 24: Demand variation

Q19:: Does the process have variation in demand?
A: Quite a lot; Just a little; No
D: Every business has to deal with a variation in the amount of demand of its products or services. A supermarket can’t predict the exact amount of beer they are going to sell on a single day, a doctor can’t predict when patients are going to visit, a fireman can’t predict when there is going to be a fire. In investment management, variation in demand is usually determined by the market: some days just a few trades happen, on other days a lot of trades are necessary. And the market affects the whole organization: If the traders trade a lot, settlements has to settle a lot, reconciliation reconciles a lot, etc. Demand variation is generally a bad thing, but it is unavoidable. Some companies try to reduce the variation in demand, for example the Dutch railways. By providing a discount card they try to encourage people to use the train in the low traffic hours. Variation in demand is usually noticed as “a busy day” or “a quiet day”.

38
R: Variation in demand is a common cause for sudden delays in processes. Increased demand will increase the utilization of the resources. An often used relation between utilization and lead time is the Erlang distribution (Brand & Kolk, 1995). That means that the lead time will increase dramatically when utilization approaches 100%.
I: The demand variation is the variance over the number of incoming cases per time unit. This can be extracted from the logs automatically

Q20: Have there been any efforts to reduce the amount of variation in demand?
A: Yes, the options were implemented/ are being implemented; Yes, but there were no possibilities; No
D: Has any serious thought been given to the reduction of demand variability? This might be solutions like actively assigning priorities to cases, and handle the most urgent once first, leaving the less important ones, the aforementioned discounts for quiet times, etc.
R: In most cases reducing variability of demand will have a positive effect, because the number of resources needed is more stable, queues are shorter and overtime and idle time are less common. See Factory Physics by Hopp and Spearman (Hopp & Spearman, 2000).
I: Knowledge

Q21: How does variation in demand affect the rest of the process?
A: A large demand will require overtime from workers; A large demand will increase the lead time; Variations in demand are not really noticeable
D: A large variation in demand will have effect on the performance of the process. Possible options may include employees having to work overtime or
R: The goal of this question is to determine the type of buffering currently used in this process.
I: Knowledge, or by plotting the demand variation over other metrics, like utilization or lead time.
5.1.1.5 Exceptions

Q22: Do exceptions occur more than the norm?
A: A bit more; A lot more; Exceptions are under the norm; Exceptions are not registered
D: A case that takes an exceptional, but modelled, route through the process, for example if an automated task fails and a case is handled by a human.
An exception is not defined as a case that takes a not modelled route through the process because something so exceptional happens that the modeller did not foresee or which happens too infrequent to model.
R: The most common reason for exceptions will be cases where an automatic system cannot handle a case and so it is passed on to a human. This is both more expensive and more time consuming.
I: Logs, list of “exceptional” functions. Count all cases that pass an exceptional function. This is automatable. See Exceptions.

Q23: Is it known how much these exceptions cost in lead time?
A: Yes; Approximately; No
D: Knowing the cost of the exceptions is the start of deciding what to you about it. Is there any knowledge about the costs (for example, the extra amount of lead time or man hours it costs?)

R: Reducing exceptions can be solved by more sophisticated software or better training. Both these improvements will cost money. Getting an idea of the actual cost of exceptions is the first step.

I: Logs. If there is a list of exceptional and not exceptional cases, these sets can be compared on properties such as lead time.

Q24: **Is it known how much these exceptions cost in service time?**

A: Yes; Approximately; No

D: Knowing the cost of the exceptions is the start of deciding what to you about it. What do exceptions cost in service time?

R: Reducing exceptions can be solved by more sophisticated software or better training. Both these improvements will cost money. Getting an idea of the actual cost of exceptions is the first step.

I: Logs. If there is a list of exceptional and not exceptional cases, these sets can be compared on properties such as lead time.

5.1.1.6 General process practices

![Diagram](image-url)
Q25: Does the process contain one or more knock-outs?
A: One; Multiple; None
D: A knock-out is a decision where a case is “Knocked out”. Usually it is a decision with one path where a lot will happen, and one path where almost nothing happens. For example during the process of requesting a loan, there is a check on ones credit rating. If that is insufficient, the case is knocked out and not processed anymore (except a couple of small tasks, like writing a rejection letter and archiving). Does this process contain such a decision?
R: There are several best practices related to knockouts.
I: The general structure (a decision) can be derived from the process model. Whether it is a knock-out or just a regular decision can be analysed using logs (if there is a large difference in lead time between path A and path B). An expert will also know this.

Q26: Are all the knock-outs in optimal formation?
A: Yes; Unknown; No
D: A knock-out is a decision where a case is “Knocked out”. Usually it is a decision with one path where a lot will happen, and one path where almost nothing happens. Sometimes a number of checks are done on a single case. It is usually best to order tasks in such a way that the task with the largest chance to “knock out” and the least effort is put first, because if a task is knocked out subsequent tasks don’t have to be done anymore.

Figure 27: Knockout example
In this example it would be best if Check 1 is easier and/or has a higher knock-out chance than Check 2.
R: Ordering knock-outs in a correct way is a safe and easy way to optimize a process, it should be considered for each process (Aalst & Hee, 2002).
I: Identical to the previous question.

Q27: Are all tasks for the knock-out required for the knockout?
A: Yes, No
D: Is the result of each task required for the execution of the knock-out?
R: If the results of a task are not required for a knock-out, it should be moved to the back. That way the execution of the task may be removed.
I: Identical to previous question.

Q28: Are there any tasks that can be put in parallel?
A: Yes, but sequential is better in this case; No; Unknown;
D: Sometimes tasks are put in sequence when there is no real good reason to do this. The reason for this is mainly that in the past a case was represented by a physical object (a file), which can be at only one place at a time. However, since a case is now a digital object in a database, it can be used in two places at once. If two tasks are in sequence, and the second one does not use any results from the first task, they could be put in parallel. This might reduce
the total lead time. However, if two different people execute the two tasks, they both have to familiarize themselves with the case. If the start-up time is a lot compared to the actual execution time, the service time and lead time might actually increase and parallelism might be a bad idea.

R: It is generally a good idea to put tasks in parallel (Aalst & Hee, 2002). This will reduce the lead time. Looking for possibilities of parallelism is something that involves a detailed knowledge of the work and the data involved in each task. The best strategy is probably to find someone who knows the process very well, explain parallelism and its effects and let him sit for an hour looking specifically and only for opportunities for parallelism.

I: If some of the first function’s output is required input for a second function, those functions cannot be put in parallel. If for each function a complete set of input and output data is available, opportunities for parallelism can be calculated. Whether the parallel situation is actually faster than the original situation can be determined by comparing the logs before and after the transformation.

5.1.1.7 Incidents

Q29: How detailed is the incident reporting?
A: The standard incident report policy is used; Smaller amounts of money are also investigated; Near misses of small amounts are also investigated.
D: Near misses are situations that could have led to an actual incident, but did not due to alertness or experience of the people involved, even though it is not their task. For example a patient who discovers the pills he got are different than his usual ones, a passenger who alerts the flight crew about a defect, etc.

There is the standard incident policy, but even smaller incidents can be interesting from a prevention point of view. What happens to those?

R: As described in section 3.3, it is interesting not only to look into the real important incidents, but also into smaller incidents and near misses. We assume that the normal incident policy is upheld.
I: Expert knowledge. This question is about the unreported, small incidents and mistakes. They are usually not reported in an official system.

**Q30: What is the most common reason that incidents are not prevented?**

A: There was a measure in place to prevent this error, but it did not function; There was no measure in place to prevent this kind of error; Unknown

D: If a mistake is made, this creates a dangerous situation. This does not have to be a problem, since there are defences against errors. These are necessary, since in every organization mistakes are made, and it is impossible to prevent them completely. Defences can be of technical nature or of organizational nature. Examples of a technological defence are warning lights in a car, checks in software packages, fuses in a fuse box, etc. Examples of organizational defences are checklists, people whose task it is to perform checks, etc. If something really goes wrong, that can mean two things. Either the defences were not present, or they did not function. What is usually the case?

R: As described in the model, if a developing incident happens either there was no planned recovery, or the recovery failed.

I: Expert knowledge.

**Q31: How many incidents do occur?**

A: Too many; Not too many

D: Is the number of incidents higher than is acceptable?

R: It is trivial that a process with a lot of incidents is probably performing worse than a process without incidents.

I: The incident reports. This would require that incidents are specified to individual processes. Could be automated by counting the number of incidents in a detailed enough set of reports,
5.1.1.8 Process quality

Figure 29: Realism of the models

Q32: Do the models in ARIS correspond with reality?
A: Cases that take a different route than is possible in the model are common; Cases that are impossible are very rare; Unknown.
D: ING IM models all processes in the tool ARIS. Ideally every model should be exactly like reality; however, this is not always the case. It might happen that there are cases which have to take a route which is not possible in the model.
R: Especially if any other, formal analysis is done on the models, the model quality should be known.
I: If the logs of a case do not correspond to the model, or if the logs of a case are absent, it indicates that the case in reality took a different route than is possible in the model. If only a very low number of cases do not correspond with the model, it was probably an exceptional case. If a high number of cases do not correspond, the model probably misses some common routes.
Q33: What % of the process is automated or supported by a system?
A: >90%; >75%; >25%; >=0%; Unknown
D: How many of the steps involved in this process, are fully automated or supported, by a specific application.
R: The amount of automated systems is a measure for the maturity of the process.
I: Logs. Most automatic systems will produce logs, or can be configured to do that.

5.1.1.9 Queue

Figure 30: Queues

Q34: If a case is not being worked on, why is that?
A: Until a certain moment; until a resource can pick it up; for completion of a batch; Waiting for a parallel branch to process; Allocated to the wrong person; Ignored or forgotten by an external party; Other/Unknown.
D: While a case is within the organization, there is not always someone actually working on it. In fact, in most organizations a case spends far more time not being worked on than being
worked on. Instead it is laying around waiting in mailboxes, to-do lists, etc. If you want to reduce the total lead time, it is important to look at this waiting time, since it is easier to reduce this than to make people actually work faster.

**R:** It is quite difficult to improve the speed people are working on a case. Two ways to improve that are training and improved technology. Those options are expensive and have practical limits. A third way is to have people specialize on one type of task. But the best chances lay in the time that a case is not being worked on. In most situations this is very high. Even percentages of 99% are not uncommon. There are different reasons a case is not worked on, the goal of this question is to determine what reasons there are in this process.

**I:** Expert knowledge. Some answers (if the case is waiting for a resource, batch work or a parallel branch) can be calculated using logs. This is not automatable, but some automated suggestions are possible.

**Q35: How is allocation to the wrong person prevented?**

**A:** Not/Informal, Each case has a designated responsible person, Formal reminder policy; A person is responsible for each case and takes care of this.

**D:** In what way are wrongful assignments prevented or handled?

**R:** There should be methods in place to solve this kind of error.

**I:** Expert knowledge. These policies are known to people in the process.
Figure 31: Queues

Q36: What is the average queue rate?
A: High; Average; Low

D: What percentage of the total time a case is in the process, is the case waiting for resources to become available?
R: The queue rate is the time cases spent waiting for resources to become available, compared to the time cases are actually worked on. A very high queue rate is considered a sign of a bad process (Davenport, 1993). This question is difficult to answer for process owners, since these numbers are nearly impossible to calculate if the process is not supported by a workflow management system.
I: Logs. This is automatable. Shown in chapter 6.

Q37: What are the most common types of tasks with large queues?
A: A normal task; Authorization task; A decision has to be made; A control
D: There are several fundamental different task types in a process. A “normal” task is a regular task where resources add value to the product. A sign task is a task where a manager has to authorize or sign for a case. A decision task is a task where a decision has to be made by a manager or senior resource. A check is a task where a resource checks whether a case satisfies some condition.

R: The type of task determines the kind of strategy that can be used to improve the process. A decision might be moved to the workers, authorization can wait if the probability of rejection is small.

I: Expert knowledge.

5.1.1.10 Utilization time

Figure 32: Utilization

Q38: Does it ever happen that resources are sitting idle because no work is available?
A: No; Very rarely; Sometimes
D: Does it ever happen that resources have nothing to do? This includes people working very slow because it’s a quiet day anyway (spending more time discussing the weekend, taking an extensive lunch break, etc) or keeping busy with things that are nice but not really necessary.
R: It is obvious that resources that are sitting idle are not cost efficient. In administrative processes, with the resources being people, this will usually be rare, since people can adapt to the available workload, something machines do not. However, it is interesting to know if it happens.
I: This can be calculated from the logs, by calculating the amount of time the resource is not working on a case, and no work is available. However, this assumes that when a resource is not logged to be doing an activity, he is doing nothing. This is a dangerous assumption, since most people have work outside of the automated systems. Expert knowledge will be required.

Q39: Why is there no work available for the resources?
A: No demand/work from customers; Bottleneck within own process; Waiting for results of another party
D: Why is there no work? In general there are three situations possible. The work can not arrive at the beginning of the process. This is because it is held up somewhere else in the organization or there is no demand at all. The work can also be stuck behind a bottleneck somewhere in the current process. Besides that it is also possible that the work is given to another party (internal or external) within the process and a result has to be awaited.

R: It is important to determine the bottleneck.

I: Bottlenecks within the own process can be detected with logs by calculating the queue size in front of each function. Variations in demand can be detected by analysing the arrival times in the beginning of the process.

Q40: Is overtime common?
A: Yes, happens every week; Sometimes, about once a month; No, rarely ever happens

D: Is it every necessary to work with more than 100% capacity to get the job done?
R: Overtime costs money and is an indication of a lot of demand variability.
I: A resource utilization of more than 100% and irregular times can be detected from the logs. This can be automated.

5.1.1.11 Start-up-time

Figure 33: Start-up time

Q41: What is the amount of unnecessary handovers?
A: A lot; Average; Low; Unknown
D: A handover is a situation where a case is handed over from one person to another person. In many situations it is best to reduce the number of handovers, because every time the case is handed over the resource will need to familiarize himself with the case.
R: Unnecessary handovers will increase the start-up time. This is a type of running cost that is quite easy to avoid, by changing the way cases are assigned.
I: This is automatale using process logs. This is detailed in chapter 6.3.2.

5.1.1.12 Workflow

Figure 34: General questions about the workflow

Q42: How is determined who works on a case?
A: Cases are picked up by the resources, Cases are assigned to resources
D: Usually there are more resources that can do a task on a case. The policy to select the resource can have influence on the process.
R: There are two general strategies, push and pull. With push, the cases are pushed to certain resources. The advantages are that the process owner is in total control and that an optimal strategy can be used. The disadvantage is that resources generally do not like it and that if a case is assigned to an unavailable person, the case might delay dramatically.
I: Expert knowledge The policy used to assign cases is know by the process owner. Otherwise it is easy to observe.

Q43: Is the process supported by a workflow management system?
A: Completely; Parts of it; not at all
D: A real workflow management system is a system where the complete process is modelled, and the system determines the movement of cases through the process.
R: If there is indeed a real workflow system, making changes in the process can become very simple. Thus the system would be really flexible.
I: Expert knowledge. Not possible to automate.

5.1.1.13 Batch

Q44: What are the reasons to use a batch?
A: Unknown; Requirement by external party; Other/Unknown; All cases over a certain period are decided at one time; An automatic job is run at a certain time; All cases are collected until (e.g.) 16:00, and then send to another department;
D: Sometimes cases are not handled individually but instead in batches. What is the reason that a batch is used?
R: In some cases batches should be removed when possible, because the introduce unnecessary wait time. They are often present because of historic reasons, for example a case that is transferred to another place. Because of postage, the cases were collected and send in packages of at least 25 cases. Since the rise of electronic and effectively free communication, a constraint like this is not required anymore, and will only slow down the process.
I: Expert knowledge. Cannot be automated
5.2 Conclusions
From the sub conclusions that have been derived in the previous sections final conclusions are generated. These are pictured in red blocks. The descriptions of the red blocks have the following format:

- N: The name of the final conclusion
- D: The description. This is a larger description that the user will see. It tells why the conclusion has been drawn, and gives solutions and examples. This is the most important results for the end user.

The examples are usually not related to investment management. This is because the risk of mistakes is too large, and the simple abstractions of the processes will only distract the experts that will be reading these texts.
Figure 36: The implemented suggestions

N: Explain problems with informal communications
**D:** Internal communication is an interesting measure for process health. According to Hammer (Hammer & Champy, 1993), too much internal communication is an indication that the process is unhealthy. Informal communication can either mean that the processes are divided in an unnatural way, or that the quality of the input is insufficient. In an ideal situation informal communication would not be necessary; the process should only communicate through a few formal channels with the outside world. This will increase speed and reduces cost (informal communication costs time and increases risks for mistakes), and increases flexibility for process change, because if there are fewer contact moments with the world out of the process, it is easier to change the process without risking breaking anything else in the organization.

**N:** Explain case-based work

**D:** Case-based work is the removal of batch-processing and periodic activities from a workflow. Sometimes work is gathered in batches or gathered until a certain time to process all cases at once. If a case arrives just after the batch has been handled, it will be waiting idle for a long time.

While in industry cases have to be handled in batches (for example if products have to bake in an oven), in administrative organizations only one case is worked on at a time. Reasons that things still are in batches can be that in the past items had to be send by post or courier, all cases are decided on a weekly meeting or computer systems which process cases at a certain times. Of course there can also be external reasons, for example relating to the stock market. Getting rid of these constraints may significantly speed up the handling of cases. Solutions may include sending information electronically and eliminating physical transport related batches, increasing the number of times the automatic scripts are run, increase the number of decision meetings or letting individual people decide over the simple cases themselves and deciding only about the hard cases on the meetings.

**N:** Apply contact reduction BP

**D:** Communication steps can have a lot of bad influence on the process. They cost time (a case that is waiting on the reaction of an external party), money (the time people are busy with communicating), quality (communication errors) and reduce flexibility (the more interconnected a process is with the outside world, the more difficult it is to change anything). Reducing the amount of communication can be a good thing. In (Jansen-Vullers & Reijers, 2005) a case is described where a family doctor referred a patient to a specialist (communication step one) and after intake the specialist decided if it was necessary to request the patient’s medical file from the family doctor (communication step two). Analysis showed that the medical file was asked for in 95% of the cases. In a redesigned process they asked for the medical file immediately at the first communication step. While this increased service time slightly (they requested the medical file even when it was unnecessary), it eliminated the second communication step, and thus sped up the whole process.

**N:** Introduce time triage

**D:** A triage is a separate path for a case, depending on the nature of the case. For example, the following case of a procedure for building permits in a town (Aalst & Hee, 2002). There was one procedure for all cases. A small renovation to a home had to go through the same six month procedure as a large office building. By splitting cases on difficulty and impact, they could realize large reductions on simple cases. This increased the customer satisfaction. The risk of a triage is that too much fragmentation in different paths may reduce flexibility, and that the lead time of the lengthy cases might increase slightly. Other variants may be: Letting
managers decide only about difficult cases, and let the workers make decisions about easier cases. Doing lengthy checks only on cases involving a minimum amount of money.

**N: Consider buffering strategy**
**D:** Every business has to deal with a variation in the demand of its products or services. A supermarket can’t predict the exact amount of beer they are going to sell on a single day, a doctor can’t predict when patients are going to visit, a fireman can’t predict when there is going to be a fire. In investment management, variation in demand is usually determined by the market: some days just a few trades happen, on other days a lot of trades are necessary. And the market affects the whole organization: If the traders trade a lot, settlements has to settle a lot, reconciliation reconciles a lot, etc. Demand variation is generally a bad thing, but it is unavoidable. Some companies try to reduce the variation in demand, for example the Dutch railways. By providing a discount card they try to encourage people to use the train in the low traffic hours. Variation in demand is usually noticed as “a busy day” or “a quiet day”.

**N: Risk of being too specialized**
**D:** If different types of cases require different roles, there is a risk of overspecialization (Netjes, Reijers, & van der Aalst, 2005). For example, in a restaurant there are two chefs, one who is very good in cooking fish dishes, and the other who is very good in cooking meat dishes. If one day, for a specific reason or just because of bad luck, there are only guests who order fish, the fish chef will be overburdened, and the meat chef has nothing to do. Solutions could be to cross train (both chefs can do everything), or at cross train at a limited level (the fish chef can do easy meat related jobs).

**N: Task elimination BP**
**D:** Task elimination is the removal of tasks from the process that are not necessary. This is most common for control tasks, but can also occur for authorization tasks. A control task is usually there for sake of security, to recover from problems in earlier tasks. Eliminating a control task will reduce cost and reduce execution time, as well as queue time for that task. However, it might have a deteriorating effect on the quality of the service. There are several possibilities to consider: A check is done on work done within this process. For each control task, the previous tasks should be examined carefully. Is it possible to guarantee correctness within the task itself? Can multiple checks be combined into one? A check is done on information done within ING IM. Incoming information should have a formal specification. Not only the format but also the correctness should be specified. For example, the checks that have been done, the laws and procedures that apply, etc. Checks should only be done on only one place. If the reason that it is done twice is that the correctness of the first check is not trusted, the first check should be improved instead of adding a second check.

**N: Empower best practice**
**D:** In some processes time is spend on authorizing work that is done by others. When workers are empowered to take decisions independently, it may result in smoother operations with lower throughput times. A drawback is that this may result in a reduction of quality, since some errors are no longer found. Other variations that solve this may include: peer reviewing, letting workers decide on easier cases or cases with a lower value, letting workers decide on cases which are due soon, etc.

**N: Parallelization of authorization/ checks**
D: If considerable delays are due to checks or authorizations, but eliminating those is not an option, they may be put in parallel to the tasks after check/authorization. That means that the total lead time can be considerable faster, but it also means that if the check fails, some of the parallel work might have been in vain. There are two requirements: The check/authorization must have a high success rate (>90%) and the work after the check must be reversible. The success rate is needed because the work after the check is thrown away if the check fails. If the success rate is too low, too much work is thrown away. The work must also be reversible, for cases that are thrown away. This means that the case is still within ING IM and that no orders or information have been sent to third parties.

N: Explain case time distribution knowledge
D: To reduce the total lead time it is necessary to know how a case actually spends its time. It can be measured by taking a number of cases, and following them in detail. In general there are three types of time for a case: service time, queue time and wait time. Service time is the time a case is being worked on. Queue time is time a case is waiting because no resources are available. This usually means that the person who handles this case is busy with another case of the same type. Wait time is time a case cannot be worked on because it is waiting on something else. For example waiting for the fulfilment of a batch or for a certain periodic event. This also includes delays due to errors, such as wrong assignment, a case being lost, etc.

N: Case manager
D: A case-manager is one person who is responsible for the handling of each case. This does not mean that he should be the only person working on the case. The case manager’s goal is to improve the external quality of the workflow, by providing a single point of contact for the customer. A case manager can also reduce the amount of errors and decrease lead time since there is clearly someone responsible.

N: Explain output definition
D: Part of a good process should be a good definition of the output. If you ever wanted to sell the service you deliver, you should make clear what you are producing. Clear output definitions should include format, conditions and timing information. Having clear output definitions has a number of advantages: The other party has a clear input definition. Increased flexibility: as long as you guarantee the output, it doesn’t matter what happens inside the process. It is easier to control the quality of your work since it is exactly known what to check. It is easier for the next party to eliminate useless control tasks, since they know what checks have been done. It is clear what adds value for your customers and what not. This helps for using customer oriented methodologies (such as 6 sigma).

N: Explain input definition
D: Part of a good process should be a good definition of the input. If you ever needed to buy the service you need, you should make clear what you are requesting. Clear input definitions should include format, conditions and timing information. Having clear input definitions has a number of advantages: Increased flexibility: as long as you can guarantee that you can handle that input, it doesn’t matter what you do afterwards. It is easier to control the quality of your work since it is exactly known what you can expect from the data you get.

N: Increase check threshold
Since it is indicated that both a lot of errors reach the end of the process, and that few iterations are done. This could indicate that the threshold of the checks is not high enough. By including more checks and making existing checks more thorough, quality can be increased.

### 5.3 Scores

The scores are also described in ARIS diagrams, as specified in chapter 4. Because these figures are rather large and unwieldy, they can be found in appendix.

<table>
<thead>
<tr>
<th><strong>Cost</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Running cost</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Good</strong></td>
<td><strong>Bad</strong></td>
</tr>
<tr>
<td>Low level of informal communication</td>
<td>High level of informal communication</td>
</tr>
<tr>
<td>Exceptions do not cost a lot of service time</td>
<td>Low utilization</td>
</tr>
<tr>
<td>No knockout possible</td>
<td>Exceptions cost a lot of service time</td>
</tr>
<tr>
<td></td>
<td>High level of formal contact</td>
</tr>
<tr>
<td></td>
<td>Case managers are used</td>
</tr>
</tbody>
</table>

**Idle time**

<table>
<thead>
<tr>
<th><strong>Good</strong></th>
<th><strong>Bad</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle time does not occur</td>
<td>Idle time does occur</td>
</tr>
</tbody>
</table>

**Start-up cost**

<table>
<thead>
<tr>
<th><strong>Good</strong></th>
<th><strong>Bad</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Low handovers</td>
<td>High handovers</td>
</tr>
</tbody>
</table>

**Table 1: Cost score**

<table>
<thead>
<tr>
<th><strong>Flexibility</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Volume flexibility</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Good</strong></td>
<td><strong>Bad</strong></td>
</tr>
<tr>
<td>Low Utilization</td>
<td>A lot of demand variation</td>
</tr>
<tr>
<td>No demand variation</td>
<td></td>
</tr>
</tbody>
</table>

**Process change**

<table>
<thead>
<tr>
<th><strong>Good</strong></th>
<th><strong>Bad</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Model quality high</td>
<td>Model quality low</td>
</tr>
<tr>
<td>Complete workflow management system</td>
<td></td>
</tr>
</tbody>
</table>

**Mix flexibility**

<table>
<thead>
<tr>
<th><strong>Good</strong></th>
<th><strong>Bad</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pull strategy</td>
<td>Variable proportions of cases</td>
</tr>
<tr>
<td>Stable proportions of cases</td>
<td></td>
</tr>
<tr>
<td>Minor variable portions of cases</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2: Flexibility score**
Quality

Output quality

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good output definition</td>
<td>High number of incidents</td>
</tr>
<tr>
<td>Low number of incidents</td>
<td>High level of informal communication</td>
</tr>
<tr>
<td>Exceptions high</td>
<td>No case manager used</td>
</tr>
<tr>
<td>Case managers are used</td>
<td>Customer discontent: Too many errors</td>
</tr>
<tr>
<td>Good customer satisfaction</td>
<td>Bad input definition</td>
</tr>
<tr>
<td></td>
<td>Bad output definition</td>
</tr>
<tr>
<td></td>
<td>Bad customer satisfaction</td>
</tr>
</tbody>
</table>

Process quality

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimal level of formal contact</td>
<td>Model quality low</td>
</tr>
<tr>
<td>Model quality high</td>
<td>Bad output definition</td>
</tr>
<tr>
<td>Good output definition</td>
<td>Bad input definition</td>
</tr>
<tr>
<td>Customers well known</td>
<td>Exceptions not registered</td>
</tr>
<tr>
<td>Good input definition</td>
<td>High level of informal communication</td>
</tr>
<tr>
<td>Low level of informal communication</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Quality score

Time

Queue time

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low queue rate</td>
<td>High queue rate</td>
</tr>
<tr>
<td>Parallelism good</td>
<td>Pull strategy</td>
</tr>
<tr>
<td>Push strategy</td>
<td>Customer discontent: Too slow</td>
</tr>
<tr>
<td>Exceptions low</td>
<td>Parallelism might be possible</td>
</tr>
<tr>
<td></td>
<td>Exceptions high</td>
</tr>
</tbody>
</table>

Service time

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exceptions do not cost a lot of service time</td>
<td>Exceptions cost a lot of service time</td>
</tr>
<tr>
<td>Low level of informal communication</td>
<td>High level of informal communication</td>
</tr>
<tr>
<td>Minimal level of formal contact</td>
<td>High level of formal contact</td>
</tr>
<tr>
<td>No knockout possible</td>
<td></td>
</tr>
</tbody>
</table>

Wait time

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case managers are used</td>
<td>Batch work</td>
</tr>
<tr>
<td></td>
<td>Delay due to misallocation</td>
</tr>
<tr>
<td></td>
<td>Suppliers are slow</td>
</tr>
<tr>
<td></td>
<td>Cases get lost</td>
</tr>
<tr>
<td></td>
<td>Parallel wait time</td>
</tr>
</tbody>
</table>

Table 4: Time score

5.4 Use cases

The tool was tested on two different processes. For each process the questions were answered by a domain expert and filled into the tool.
As explained in chapter 4, each metric gets a score on a three level scale: Green (with a +), Yellow and Red (with a −). For the lowest level scores (the scores indented most to the right)
a green ball means that this sub conclusion is positive for this particular aspect, a red ball means it is negative. The metrics aggregate the different sub conclusions. If a metric is in the highest 33% of a possible score, it will get green ball. A red ball if the score is in the lowest 33% and a yellow ball if it is in between.

**Change management.**

Change management is the process which handles changes within other processes. The final score was:

![The score overview of the change management process](image)

These scores were not surprising. This process often takes a lot of time, with a lot of communication (both formal and informal). Quality however is good, due to a high customer satisfaction and good definitions of all input and output. Flexibility is bad due to high
variation in demand (resulting in low mix flexibility). Cost is average with some good aspects and some bad aspects.

**Settlements:**

The settlements process is the process where the payment for a trade is arranged. The scores where the following:

![Score overview of the settlements process](image)

With 3 yellow and one red score this process is just below the middle of the spectrum. Whether that is really a bad thing would depend on scores of other processes: it might not be the case that all processes average out on yellow. Another reason for the yellow scores might be that for some questions a definitive answer could not be given due to lack of information. In that case the tool considers an answer as neither good nor bad, and will result in a yellow score.
Conclusion
From this very small test the conclusion is that according to this method, settlements is the worst performing process of the two and should be looked into first for an improvement project. However, to get a good view of the quality of a process many other processes should be tested. This should not be too hard. Depending on the familiarity of the domain expert with several of the terms used, filling in the program could take anywhere between 30 minutes and an hour. Without investing a lot of time a general overview of the process landscape could be obtained using this method. This should give ideas about where to start more detailed improvement projects.
6 Partial automation of the checklist

The questionnaires of the previous chapter only use the knowledge of domain experts as input. However, for both accuracy as well as speed, the usage of more solid data is preferred. The following section assumes the availability of computer readable process models and log files. The goal is to answer some questions of the model automatically based on actual facts, instead of domain knowledge. When selected questions occur in the model, the tool will give the user the option to load log files, and calculate an analysis.

In the previous chapter it was indicated per question if it was possible to automate. Of those questions a number was selected for implementation:

- Q23: Do exceptions occur more than the norm?
- Q25: Is it known how much these exceptions cost in lead time?
- Q38: What is the average queue rate?
- Q43: What is the amount of unnecessary handovers?

The reason these questions were selected are that queue rate, exceptions and handovers are important factors and that only logs are required for their calculation. Moreover, any improvements related to handovers are often realised by modifying policies, and thus are easily implemented.

All these questions return some specific numerical value. It is assumed that there are norms available on what is acceptable and what is not. These norms can be produced by domain experts, from comparable processes in the market, or from averages of the own process over a period of time.

<table>
<thead>
<tr>
<th>Question</th>
<th>Automation</th>
<th>Question</th>
<th>Automation</th>
<th>Question</th>
<th>Automation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>A</td>
<td>Q17</td>
<td></td>
<td>Q33</td>
<td>X</td>
</tr>
<tr>
<td>Q2</td>
<td>A</td>
<td>Q18</td>
<td>X</td>
<td>Q34</td>
<td>X</td>
</tr>
<tr>
<td>Q3</td>
<td>X</td>
<td>Q19</td>
<td>X</td>
<td>Q35</td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td></td>
<td>Q20</td>
<td></td>
<td>Q36</td>
<td>X</td>
</tr>
<tr>
<td>Q5</td>
<td></td>
<td>Q21</td>
<td>A</td>
<td>Q37</td>
<td></td>
</tr>
<tr>
<td>Q6</td>
<td></td>
<td>Q22</td>
<td>X</td>
<td>Q38</td>
<td></td>
</tr>
<tr>
<td>Q7</td>
<td></td>
<td>Q23</td>
<td>A</td>
<td>Q39</td>
<td>X</td>
</tr>
<tr>
<td>Q8</td>
<td></td>
<td>Q24</td>
<td>X</td>
<td>Q40</td>
<td>X</td>
</tr>
<tr>
<td>Q9</td>
<td></td>
<td>Q25</td>
<td>X</td>
<td>Q41</td>
<td>X</td>
</tr>
<tr>
<td>Q10</td>
<td>A</td>
<td>Q26</td>
<td>X</td>
<td>Q42</td>
<td></td>
</tr>
<tr>
<td>Q11</td>
<td></td>
<td>Q27</td>
<td>X</td>
<td>Q43</td>
<td></td>
</tr>
<tr>
<td>Q12</td>
<td></td>
<td>Q28</td>
<td>X</td>
<td>Q44</td>
<td></td>
</tr>
<tr>
<td>Q13</td>
<td></td>
<td>Q29</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q14</td>
<td></td>
<td>Q30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q15</td>
<td></td>
<td>Q31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q16</td>
<td></td>
<td>Q32</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Overview of the questions

In Table 5: Overview of the questions are all questions, and whether they can be automated. An X means that automation is possible. An A means that automation can be used to provide the domain expert with input, but that the domain expert has to answer the question. The bold questions are questions of which the automation is implemented. They are described further in
In this chapter. The italic questions are questions that can be calculated using the replay strategy explained in this chapter. They are not implemented.

In the first part the syntax and a semantic of EPC’s will be given. In the second part additional definitions and metrics will be explained. In the third part the metrics and in the fourth part the algorithm is given.

6.1 Definition of EPCs

6.1.1 Syntax

The subsequent syntax definition is from (Mendling, 2008)

A flat EPC = (E, F, P, C, l, A) consists of four pair wise disjoint and finite sets E, F, P, C, a mapping l: C → {and, or, xor} and a binary relation A ⊆ (E ∪ F ∪ P ∪ C) × (E ∪ F ∪ P ∪ C) such that

- An element of E is called event, E ≠ ∅
- An element of F is called function. F ≠ ∅
- An element of P is called process interface
- An element of C is called connector
- The mapping l specifies the type of a connector c ∈ C as and, or, or xor.
- A defines the control flow as a coherent, directed graph. An element of A is called an arc. An element of the union N = E ∪ F ∪ P ∪ C is called a node.

In order to allow for a more concise characterization of EPC’s, notations are introduced for preset and postset nodes, incoming and outgoing arcs, paths, transitive closure, corona and several subsets.

Preset and postset of nodes: Let N be a set of nodes and A ⊆ N × N a binary relation of N defining the arcs. For each node n ∈ N, we define • n = {x ∈ N|(x, n) ∈ A} as its preset and n • = {x ∈ N|(n, x) ∈ A} as its postset.

Incoming and outgoing arcs: Let N be a set of nodes and A ⊆ N × N a binary relation over N defining the arcs. For each node n ∈ N, we define the set of incoming arcs n_in = {(x, n)|x ∈ N ∧ (x, n) ∈ A}, and the set of outgoing arcs n_out = {(n, y)|y ∈ N ∧ (n, y) ∈ A}.

Paths and connector chains: Let EPC = (E, F, P, C, l, A) be a flat EPC and a, b ∈ N be two of its nodes. A path a ↦ b refers to the existence of a sequence of EPC nodes n_1, ..., n_k ∈ N with a = n_1 and b = n_k such that for all i ∈ 1, ..., k holds:

(n_1, n_2), ..., (n_i, n_{i+1}), ..., (n_{k-1}, n_k) ∈ A. This includes the empty path of length zero, i.e., for any node a: a ↦ a. If a ≠ b ∈ N and n_2, ..., n_{k-1} ∈ C the path a ↦ b is called a connector chain.

Upper Corona, Lower Corona: Let EPC = (E, F, P, C, l, A) be a flat EPC and N its set of nodes. Then its upper corona is defined as * n = {v ∈ (E ∪ F ∪ P)|v → n} for some node n ∈ N. It includes those non-connector nodes of the transitive preset that reach n via a connector chain. In analogy, its lower corona is defined as n * = {w ∈ (E ∪ F ∪ P)|n → w}.

Subsets:

E_s = {e ∈ E| • e = 0 ∧ |e • | = 1} being the set of start events,
For an EPC to be *syntactically correct*, it must fulfill a set of requirements, such as being a directed graph and have at least one start and one end event. The complete requirements can be found in (Mendling, 2008, p. 25). Also described there is the relaxed syntactically correct EPC, which has a less strict set of requirements.

### 6.1.2 Semantics

Over the years several semantics for EPC’s have been proposed, but most semantics have limitations (Mendling, 2008). These are often related to the OR-Join, because of synchronization problems. The problem is that if there is a token on one branch of an OR-join, it is unknown whether it can fire. It can only fire if it is not possible for another token to arrive on the other branch, but this is non-trivial to determine.

I use the semantics described in (Mendling, 2008), which were first published in a (Mendling & Aalst, 2006).

The principle of these semantics is based on Boolean nets by Langner, Scheider and Wehler. The idea is the following:

They define a *state* and a *context* over all the arcs of an EPC’s.

The *state* is a mapping of positive and negative tokens over all arcs. A positive token indicates which functions can be executed, a negative token indicates that the functions can be ignored for now.

The *context* assigns a value of *wait* or *dead* to each arc. A *wait* context indicates that it is still possible that a positive token might arrive. A dead context status means that either a negative token will arrive next or no positive token can arrive anymore. The main use of the dead context is to signal an OR-Join that no positive token will arrive. In that case, when all incoming arcs of an OR-join are either positive tokens, negative tokens or have a dead context, the OR-join knows it can fire.

**Definition 2.12 (State and Context).** Let \( EPC = (E,F,C,I,A) \) be a flat EPC. Then a mapping \( \sigma : A \rightarrow \{-1,0,+1\} \) is called the state of an EPC. The positive token captures the state as it is observed from outside the process. It is represented by a black filled circle. The negative token depicted by a white open circle with a minus on it has a similar semantics as the negative token in the Boolean nets formalization. Arcs with no state tokens on them do not depict a circle. Furthermore, a mapping \( \kappa : A \rightarrow \{\text{wait}, \text{dead}\} \) is called a context of an EPC.

A *wait context* is represented by a \( \text{wait} \) and a *dead context* by a \( \text{dead} \) next to the arc.

The state and the context are together known as a *marking* of an EPC.

**Definition 2.13 (Marking of an EPC)** Let \( EPC = (E,F,C,I,A) \) be a EPC. Then, the set of all markings \( M_{EPC} \) of an EPC is called marking space with \( M_{EPC} \subseteq A \rightarrow (\{-1,0,+1\} \times \{\text{wait}, \text{dead}\}) \). A mapping \( m \in M_{EPC} \) is called a marking. Note that defines the two mappings presented above, ie, \( m(a) = (\sigma(a),\kappa(a)) \). The projection of a given marking \( m \) to a subset of arcs \( S \subseteq A \) is referred to as \( m_S \).

Each EPC has a set of initial and final markings:
Initial marking of an EPC. Let $EPC = (E, F, C, I, A)$ be a relaxed syntactically correct EPC and $M_{EPC}$ its marking space. $I_{EPC} \subseteq M_{EPC}$ is defined as the set of all possible initial markings, i.e. $m \in I_{EPC}$ if and only if:

- $\exists a_s \in A_s: \sigma_m(a_s) = +1$,
- $\forall a_s \in A_s: \sigma_m(a_s) \in \{-1, +1\}$,
- $\forall a_s \in A_s: \kappa_m(a_s) = \text{wait if } \sigma_m(a_s) = +1 \text{ and } \kappa_m(a_s) = \text{dead if } \sigma_m(a_s) = -1$, and
- $\forall a \in A_{int} \cup A_e: \kappa_m(a) = \text{wait and } \sigma_m(a) = 0$

Final Marking of an EPC. Let $EPC = (E, F, C, I, A)$ be a relaxed syntactically correct EPC and $M_{EPC}$ its marking space. $O_{EPC} \subseteq M_{EPC}$ is defined as the set of all possible final markings, i.e. $m \in O_{EPC}$ if and only if:

- $\exists a_e \in A_e: \sigma_m(a_e) = +1 \text{ and } \kappa_m(a_e) = \text{wait and}$
- $\forall a \in A_s \cup A_{int}: \sigma_m(a) \leq 0 \text{ and } \kappa_m(a) = \text{dead}$.

In four phases (dead context propagation, wait context propagation, negative token propagation and positive token propagation) all other viable markings between the initial and final markings are calculated. The rules are shown in Figure 39, which is from (Mendling, 2008). A more complete description can be found there.
Figure 39: The four types of propagation

1) Dead Context Propagation

2) Wait Context Propagation

3) Negative State Propagation

4) Positive State Propagation
6.2 Definitions required to calculate the metrics

6.2.1 Roles & Resources

• $R$ is the set of all roles
• $U$ is the set of all resources

The roles are hierarchical, and thus $R$ is a partially ordered set. There is a mapping $\rho: F \to \mathcal{P}(R)$ where $\mathcal{P}(R)$ is the powerset set of all roles. $\rho(f)$ is the set of roles that can perform function $f$. If a role is in $\rho(f)$ then all roles higher in the hierarchy are also in $\rho(f): r < r' \land r \in \rho(f) \Rightarrow r' \in \rho(f)$

Each resource has exactly one role. Should a resource have two roles ($r_a$ and $r_b$) a new virtual role ($r_{ab}$) can be created which stands in hierarchy just above both $R_a$ and $R_b$. If resources can have multiple roles it is impossible to calculate a meaningful minimum number of handovers (see below).

There is a relation $\mu: (U, R) \in \mu$, which assigns a role to each resource.

6.2.2 Logs

I assume the presence of automatic logs of the system. Logs describe events that happened, when and by whom on which case. It is assumed that one log entry is made for each time a case passes a function. In the log entry the following information is stored: the case it concerns, the function, the resource that performed the function, the time he started and the time he was finished. The following restriction is placed on the logs: it is impossible that a function is handling one case twice at the same time. This also implies that each log entry is unique, and that the total collection of logs is a set.

**Definition: Log**

Let $EPC = (E, F, C, I, A)$ be a flat EPC. A log $L$ is a set of tuples $(k, f, u, t_b, t_e)$, where $k$ is the case-id, $f$ is the function ($f \in F$), $u$ is the resource ($u \in U$), $t_b$ is the begin time and $t_e$ is the end time. One such a tuple is called a log entry. A log entry has an implicit mapping to its values. If $l$ is a log entry, $l_k$ is the case-id of that log entry, $l_f$ is the function of that log entry, etc.

All metrics are calculated per process, and per case. A trace represents the path of one case through the process. A trace is a subset of $L$ and contains all log entries regarding one case, and only entries regarding one case. This means that each case in the log has exactly one trace, and that the union of all traces equals the log.

**Definition (Trace):**

Let $EPC = (E, F, C, I, A)$ be a flat EPC and let $L$ be a Log of the EPC. A trace $T$ for case $p$ is a subset of $L$ for which the following holds: $\text{Trace } T = \{l | l \in L \land l_k = p\} \land (\forall l : l \in L \setminus T : l_k \neq p)$

Each trace has a mapping $\tau: 0..|T| - 1 \to l$. This mapping gives T ordered by $l(t_b)$ (begin time). This means that $(\forall n \in \mathbb{N}: 0 \leq n < |T| - 2 \land \tau_n = l \land t_{(n+1)} = l \land (t_b) \leq l'(t_{b}))$. The shorthand expression $T_p$ is the trace concerning case $p$.

6.3 Metrics

All metrics are calculated per trace. The end results are the averages over a set of traces.
6.3.1 **Exceptions**

An exception is defined here as a case that passes at least one function which is in a set of functions that one prefers to avoid. For example, when all cases should go straight through the process automatically, but some cases need to be handled manually, the manual function is not preferred. Another example is a function which is used to correct mistakes made earlier in the process.

In general we can assume that it is known what cases in a process are part of the preferred path and what cases are part of the exceptional path. It is therefore assumed that there is a set FE of all cases that are part of an exceptional path.

**Questions it applies to:**
- Do exceptions occur more than the norm?
- How much costs an exception in lead time?

**Implementation**

The exception rate is the percentage of cases that pass at least one function that is marked as “exceptional”:

\[
\text{Number of exceptional cases} = \{k \in K | \exists l \in T_k : l \in FE \}
\]

\[
\text{Exception rate} = \frac{|\text{Number of exceptional cases}|}{|\text{Total number of cases}|}
\]

**Figure 40: The exception question**

The norms estimated here are: 0.15 and 0.05. The tool will select “A lot more” if the exception rate is over 0.15, and “A bit more” if the exception rate is over 0.05.

Exception delay factor: The number of times an exceptional case is slower than a normal case.
Exception delay factor: \[
\frac{\text{Average lead time of exceptional case}}{\text{Average lead time of a normal case}}
\]

Exception cost factor: The number of times an exceptional case is more expensive than a normal case.

\[
\frac{\text{Average service time of exceptional case}}{\text{Average service time of a normal case}}
\]

6.3.2 **Handovers:**

A handover is a situation where a case is handed over from one person to another person. In many situations it is best to reduce the number of handovers, because every time the case is handed over the resource will need to familiarize himself with the case.

Three kind of handovers can be identified, the real handover, the required handover and the practical handover.

- The **real handover** is a handover that actually happened, based on logs.
- A **required handover** is a handover that has to happen according to the process description. That means that there are two adjacent functions where there is no user that can perform both functions.
- The **practical handover** assumes that all jobs are performed by the lowest possible roles.

In many situations the required handover gives a very low number. The most extreme case is when there is a role that is in hierarchy above all other roles (for example the boss) who is allowed to do all functions. That would mean that the minimal number of handovers is zero, since one role can do everything alone. This is far from realistic, since in a typical organization there are far more resources that are low in the hierarchy than people who are high in the hierarchy and most jobs are executed by people who are low in the hierarchy.

In these metrics only the practical handover and the real handover are used. The required handover is included in the definitions for completeness. It would not be difficult to include the required handover.

**Questions it applies to:**
- What is the amount of unnecessary handovers?

**Implementation:**

Let \( t \) be a token that moves through the process, according to the semantics described before. Assume that \( a \) and \( b \) are two functions. \( T_a \) and \( T_b \) are the two transformations of type 4-a over respectively \( a \) and \( b \) on token \( t \). \( al_u \) and \( bl_u \) are the two log entries where those function executions are recorded.

There is a real handover iff \( a \xrightarrow{c} b \) and \( al_u \neq bl_u \)

There is a practical handover iff \( a \xrightarrow{c} b \) and \( \rho(a) \cap \rho(b) = \emptyset \)

There is a required handover iff \( a \xrightarrow{c} b \) and \( \rho(a) \cap \rho(b) = \emptyset \)
Unnecessary handovers: The amount of handovers compared to the optimal situation:

Unnecessary handovers: \textit{Average number of real handovers} 
- \textit{Average number of practical handovers}

Figure 41: Handover question

\textbf{The norms}
A high number of handovers is more than 3 handovers more than required, a medium number of handovers is between 1.5 and 3 more than required and a low number of unnecessary handovers is between 0 and 1.5 more than required. These numbers are estimated.

6.3.3 \textbf{Queue time}
The distribution of times for a certain trace is an important measure for the performance of the process. As suggested before, an extremely high queue time usually is an indicator for a bad process. The time also gives a metric for other situations, for example for comparing the lead time of a case with an exception to the lead time of a case without an exception.

There are three kind of times related to a function. This first is the service time, which is the time required to do a certain job. Sometimes it is possible to decrease service time (for example, buy a better machine, train people to do their job faster), sometimes it is not possible (if a bread has to bake for two hours in an oven, it will always take two hours, no matter what you do). In these examples I assume that service time is a given and cannot be altered.

The second is queue time. Queue time is the time a case is waiting because no other resources are available. It does not include waiting for a parallel branch to complete, or because it is required for that case to wait for some time (for example when all cases are collected for transport at 16:00, or when there is a 6 weeks complaint period for certain legislation).
The third is wait time. Wait time is everything that is not service or queue time.

**Examples**

For the following definitions I will use the following simple process as an example:

![Figure 42: Queue time example](image)

Assume we have logs with the following information:

The queue time of a is 50 minutes, the service time is 10 minutes. The queue time of b is 100 minutes and the service time is 20 minutes.

**Implementation:**

**Lead time:**

The time a case takes to go entirely through the process. It is equal to the difference of the end time of the last function minus the begin time of the first function.

\[ \text{Leadtime}(k) = \text{maxtime}(k) - \text{mintime}(k) \]

where \( \text{maxtime}(k) \) is the maximum end time of the logentries concerning \( k \):

\[ \text{maxtime}(k) = l_{te} \quad \text{where} \quad l \in L_k \wedge (\forall l': l' \in L_k: l'_{te} \leq l_{te}) \]

And \( \text{mintime}(k) \) is the minimum entry time of \( k \):

\[ \text{mintime}(k) = l_{tb} \quad \text{where} \quad l \in L_k \wedge (\forall l': l' \in L_k: l'_{tb} \geq l_{tb}) \]

**Total service time:**

The total service time is the total amount of time resources are working on a case. It is dependent on the process and the route taken, but not on the actual execution. It is simply the sum of the service times of all functions. In the example above the total service time is 30 minutes.

**Total queue time:**

Queue time is the time a case is waiting because no other resources are available. In the example above the queue time is 150 minutes.

It does not include waiting for a parallel branch to complete, or because it is required for that case to wait for some time (for example when all cases are collected for transport at 16:00, or when there is a 6 weeks complaint period for certain legislation).

**Queue rate:**

The amount of time cases spent in queues.

\[ \text{Queue rate} = \frac{\text{Total queue time}}{\text{Total service time} + \text{Total queue time}} \]
6.4 The algorithm

The goal of this algorithm is to calculate the aforementioned metrics of a process. The metrics are calculated per single trace, and can be averaged to get an approximation of the metrics over the whole process. The input for the algorithm is the process and a trace and the output is the calculated metrics for each trace.

The algorithm replays every case through the process, according to the semantics described in paragraph 6.1.2. Replaying a case is necessary because using only the logs does not reveal all information. For example, if the logs of a case indicate that the case was handled by person A, B, A, B, A, B in succession, one might conclude that there are 5 handovers. However, if the process consists of two parallel paths, with each path done by one person, there are no handovers at all.

The general idea of the algorithm is the following:

The solution of a trace is calculated by constructing a tree. The nodes of the tree represent a reachable marking. The whole tree represents all possible choices for this trace. First all the possible starting markings are calculated and form the children of the root of the tree. Now a tree is constructed, where the marking represented by each node is reachable from the marking of the parent node with one step of the semantics of EPC’s. There are two propagations of positive tokens that are ambiguous. Those are the or-split and the xor-split. If one of these is encountered, the tree forks into multiple branches.
Each of these branches is handled separately. The transition of a positive token through a function (transformation 4a) is only allowed when that function is in front of the trace. The function is then removed from the trace. This way the algorithm replays the case according to the trace. If during execution, there is no possible correct move (e.g. the only function in front of a token is A, while the next function in the trace is B), it means that the correct solution is on another branch of the tree. This branch of the tree is then pruned. When the complete trace is empty and the process is in a final marking it means a correct solution has been found for this case and the metrics can be returned. If a trace yields no correct solution, the trace was not possible in the process. This trace is considered to have no value and is not used in any further calculations. Because all functions are unique (it is a set), it is not possible that one can do two different steps based on one next element in the trace.

All the metrics of a set of cases can be used to calculate the approximate values for the metrics of the whole process.

This algorithm cannot support all possible processes. In the process in Figure 44, the trace “BA” can have two possible meanings: Both in parallel, after the or-split or sequential in an iteration. This can have an effect on the metrics: the number of handovers can be different between these two scenarios (if A and B are executed by two different people, the parallel solution will have no handovers, while the sequential solution has one handover). The only way to determine the exact route would be to log all connectors in the trace. This is unlikely to happen in practice. In the current implementation of the algorithm these ambiguous traces are discarded, other solutions could be implemented.

Figure 44: A process with a ambiguous trace
6.4.1 **Definitions**

The following extension is made to the original definition of mapping $\sigma$:

The original definition for state is a mapping $\sigma: A \rightarrow \{-1, 0, +1\}$ over the arcs. The definition of state is extended to the mapping: $\sigma: A \rightarrow \{-1, 0, \text{token}\}$, to make it possible to add additional information in the tokens.

**Definition (Token).** A token represents a part of a case in a process. It is possible for one token to represent the whole case.

A token has the following fields:

- **time** – The current time of a token. It represents the last time a function was executed (transformation 4a) on this token.
- **u** – $u \in U$. The resource who executed the last function on this token.
- **roles** – The set of roles that could have executed the last function on this token.
- **minimalLeadTime** - Minimal lead time of a token. Represents the earliest time this token could have arrived on this arc, assuming that it did not have to wait for resources to become available anywhere.

It is possible that a token is the result of a merger of multiple tokens. This happens in an AND-join or an OR-join. If $TS$ is the set of tokens before the merger:

- Field **time** is the maximum time of $TS$
- Field $u$ is the intersection of $TS.u$. It only has a value if all tokens were executed by the same person.
- Field **roles** is the intersection of $TS.\text{roles}$. It only has a value if there is a role that could have executed all the previous functions.
- Field **minimalLeadTime** is the maximum of $TS.\text{minimalLeadTime}$.

A **Solution** $(Me, T, m)$ is an object which represents a possible solution for a trace. It contains the set of metrics until now, a marking and the part of the trace that has not yet been used. Solution also contains a couple of methods:

- **propagateDeadContext();**
- **propagateWaitContext();**
- **propagateNegativeTokens();**

These functions update the marking of the solution according to transformations 1, 2 and 3 respectively.

The metrics in a Solution are:

- **beginTime**
- **endTime**
- **totalServiceTime**
- **totalRunTime**
- **totalQueueTime**
- **totalWaitTime**
- **minimalLeadTime**
• realHandOvers
• practicalHandOvers
• exception

The function \textit{propagatePositiveToken(Solution s)} takes one Solution, and returns a set of solutions. The function behaves identical to the description with the following additions:

• Rule 4a (propagation through a normal function): This type of propagation is only allowed if the first element of the Trace is this function. After this transformation, the first element of the trace is removed.
• Rule 4e and rule 4g (the OR- and XOR-splits): These rules have multiple options. If these are encountered, a copy of the current solution is made and each option is explored separately. If no OR or XOR splits are encountered, \textit{propagatePositiveToken} returns a singleton set.

If a situation is encountered where further propagation is impossible, there are two possible situations:

• The marking is part of the set of final markings and the trace is empty. This means that the trace is executed correctly and that the final results can be calculated.
• The marking is not a part of the final markings, or the trace still contains elements. This means the execution of the trace was incorrect and that a wrong choice was made by a previous OR or XOR split. This solution can be discarded.

If none of the solutions end in a final state, this trace does not represent a valid path through the EPC.

6.4.2 \textbf{The algorithm:}

The algorithm takes an EPC and a trace. It returns a set of metrics for that specific trace.

1. \textbf{Require:} \textit{EPC} = (E, F, C, l, A), \textit{T} = \textit{Trace}
2. \textit{l} \leftarrow \textit{getInitialMarkings}(\textit{EPC})
3. \textit{L} \leftarrow \textit{SortByTb}(\textit{L})
4. 
5. \textbf{forall} \ (i \in \textit{l}) \ \textbf{do}
6. \ \textit{Solution} = (\textit{DefaultMetrics}, \textit{L}, i)
7. \ \textit{Solution}.beginTime \leftarrow L_{0_{tb}}
8. \ \textit{toBePropagated}.add(\textit{Solution})
9. \ \textbf{od}
10. \ \textbf{while} \ \textit{isNotEmpty}(\textit{toBePropagated}) \ \textbf{do}
11. \ \ \ \textit{CurrentSolution} \leftarrow \textit{toBePropagated}.pop()
12. \ \ \ \textit{currentMarking} = \textit{CurrentSolution}.m
13. \ \ \ \textit{CurrentSolution}.propagateDeadContext()
14. \ \ \ \textit{CurrentSolution}.propagateWaitContext()
15. \ \ \ \textit{CurrentSolution}.propagateNegativeTokens()
16. \ \ \ \textit{Solutions} \leftarrow \textit{propagatePositiveTokens}(\textit{EPC, CurrentSolution})
The function of propagatePositiveTokens is much longer, and is described in appendix 9.1.

MergeTokens merges a set of tokens after an and-join or or-join, and updates the metrics. It takes a set of tokens and returns a single token. Let TS be the set of tokens to be merged. The new token t has the following properties:

MergeTokens(Set of Tokens TS)

1. newToken.minimalLeadTime ← max(\{t.minimalLeadTime|t ∈ TS\})
2. newToken.u ← \bigcap_{t ∈ TS} t.u
3. newToken.roles ← \bigcap_{t ∈ TS} t.roles
4. newToken.time ← max(\{t.time|t ∈ TS\})
5. return newToken

Most metrics are calculated on the execution of a function \( f \). Assume that S is a solution object, with several variables representing the metrics until now. Let \( u \) be the resource who executed \( f \) according to the logs. \( t_b \) is the time resource \( u \) started working on \( f \), \( t_e \) is the time he was finished. The function \texttt{updateMetrics()} calculates the new value for all metrics.

\[
\texttt{UpdateMetrics():}
\]

1. \( S.serviceTime ← S.serviceTime + t_e - t_b \)
2. \( S.queueTime ← S.queueTime + t_b - \text{token.time} \)
3. \( S.runTime ← S.runTime + t_e - \text{token.time} \)
4. \( S.minimalLeadTime ← S.minimalLeadTime + t_e - t_b \)
5. \( S.endTime ← t_e \)
6. \( \text{if } u ≠ \text{token.user }→ \)
7. \( S.realHandOvers ← S.realHandOvers + 1 \)
8. \( \text{token.user }← u \)
9. \( \text{fi} \)
10. \( \text{if } \text{token.roles }∩ \rho(f) = \emptyset \text{ then} \)
11. \( S.requiredHandovers ← S.requiredHandovers + 1 \)
12. \( \text{fi} \)
13. \( \text{if } \perp (\text{token.roles}) ∩ \perp \rho(f) = \emptyset \text{ then} \)
14. \( S.practicalHandovers ← S.practicalHandovers + 1 \)
6.4.3 Test of the algorithm

This is a process the algorithms were tested on. It is assumed there are 4 roles: P, Q, R and U. The functions had the following properties:

<table>
<thead>
<tr>
<th>Function</th>
<th>Service time</th>
<th>Acceptable roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>Normal distribution, $\mu = 600, \sigma = 120$</td>
<td>P, Q, R</td>
</tr>
<tr>
<td>F2</td>
<td>Constant, 60</td>
<td>U</td>
</tr>
<tr>
<td>F3a</td>
<td>Exponential distribution, $\mu = 600$</td>
<td>P, Q, R</td>
</tr>
<tr>
<td>F3b</td>
<td>Normal distribution, $\mu = 600, \sigma = 120$</td>
<td>P, Q</td>
</tr>
<tr>
<td>F4</td>
<td>Exponential distribution, $\mu = 300$</td>
<td>P, Q, R</td>
</tr>
<tr>
<td>----</td>
<td>--------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>F5</td>
<td>Exponential distribution, $\mu = 720$</td>
<td>P, Q, R</td>
</tr>
</tbody>
</table>

**Table 6: Parameters of the process**

There are two connectors where a choice has to be made: A case arriving at connector a has 45% probability to go to E3a, 20% probability to go to E3b and 25% probability to go to both. Cases arriving at connector b have 95% probability to go to E6 and 5% probability to go to E2.

E2 is considered to be an exceptional function.

1000 cases were put through this process, and logged. The algorithm was executed on these logs.

**Results**

The results of the algorithms were the following on random set of logs:

- The average number of handovers was 2.32, while only 0.21 handovers where necessary on average. The program will return that the answer is “a lot of handovers”.
- Queue rate was 99.24%. This results in a score of “high” according to the norms used here.
- The exception rate is 6%, which results in an answer of “A bit more”.
- An exceptional case took 1.9 times longer than an normal case. This was considered “not too much” in the answers.

**Performance**

Even without a lot of optimization, this algorithm performs very fast. These 1000 cases can be calculated within 5 seconds. That will mean that the calculations can be used in practice while filling in the questions.
7 Conclusion

Process redesign is an interesting, but difficult problem. The literature is consistent in the fact that the first step should be to formally describe the processes. This is a logical first step, since it gives insight in a process and gives a foundation for any improvement. It is also required for any form of automation by a workflow engine.

The actual improvement process has also been explored in more detail. There should be measurements before and after the redesign, so that it can be shown that there actually was any improvement. The different kind of improvements have been categorized, for example in the best practices. And there is a lot of literature on how to implement these improvements.

The steps in between, selecting where to start and what to do, are far less clear. That is the part I have tried to explore.

It would be unrealistic to expect some method that gives unambiguous directions on what to do. It will always be necessary to setup a redesign plan, predict and calculate the consequences, and implement it. I have developed a method that will give scores to a process, indicating the current flaws in the process. This can help with prioritizing the process improvements.

I have used the devil’s quadrangle express the qualities of a process. The four aspects (quality, flexibility, time and cost) cover the most interesting properties of a process.

When scoring a process, the first issue is the availability of information. Three types of information can be identified: Formal information about the performance, informal information about the performance and information about the process itself. Formal information includes logs, reports, incidents and anything else that is measured. Informal information consists of the knowledge of the people involved. The information about the process itself are the models and associated diagrams.

One problem with the models is that they are designed with other goals than automatic improvement in mind. In the case of ING it was mainly to explain the processes to the people involved and to show compliance to regulators. An important difficulty of processes that are meant to be interpreted by humans is that information will be in natural language instead of parameters. Natural language if fine for humans, but it is difficult to interpret for a computer.

Informal information is probably the most powerful source of information about a process. Davenport already notes that serious problems are usually already known by the actors within a process. The challenge is to obtain and use that information. The solution I have chosen is a questionnaire for process owners. The questionnaire has been tried twice, iteratively improving the list. Several problems arose: It is important to be careful with the amount of academic jargon in the questionnaire. While the process owners know the process, they do not always know the theory and the official terms behind it. A lot of numbers are not known and are often estimated instead of measured.

One problem is with the scoring is the exact norms of what is acceptable and what is not. Where is the norm exactly? Queue time for example is inevitable, but too much is an indicator of a bad process. Where the threshold is, is unknown and probably different per situation. A second important problem is the weight between different factors. A high level of cases that do not go straight through the process is a bad sign, and will increase the lead time. So will a
high level of queue time, a high delay due to authorization and a high level of informal communication. But it is unclear how to measure these metrics, and even more difficult how to compare them.

The automatic calculation of metrics has a number of advantages. It is fast, it is not subjective and it is more suitable to compare different processes. The main disadvantage is that data has to be available in interpretable formats. I have developed an extension for the tool to answer some questions automatically. In this project I have tested the program with artificial values. The calculations work out, but the program has not been tested on any real process.

One problem that could arise in real world application is the difference between roles in my assumptions and roles in real situations. I assume with role hierarchy that the higher roles are allowed and capable to execute all tasks that roles below them can do. This allows for perfect modelling of junior and senior roles, where the senior role gets more permissions. However, most organizational charts will be of a type that higher in the hierarchy means “reports to” and being higher in the organization does not mean that someone can actually execute more tasks. Another possible problem is the assumption that all activities of resources are logged. If resources have tasks that are not described in the logs, something that will often be the case, it might introduce errors in the calculations. This includes resources taking a break.

In the end, I think the conclusion is that there is a lot of work to be done on this subject. There really is demand for a good applicable method to score a process. It can help correctly focusing redesign, but it can also help process owners to evaluate their own performance. When formal information is not available, a situation that will often be the case, a questionnaire using the devils quadrangle can be a viable solution. To improve this method, many empirical tests should be done to discover viable weights between metrics, discover metrics, solutions and questions that were not included, and further test its feasibility.
8 Bibliography

9 Appendix

9.1 Propagate positive tokens

A very important function is the function propagatePositiveTokens, so it is described in more detail.

The function propagatePositiveTokens takes a Solution and an EPC, recursively propagates the positive tokens until no further move can be made and updates the metrics. If a choice (XOR-split or OR-split, transformations 4e and 4g) is encountered, there are multiple options, and returns a set of Solutions. This set will contain multiple Solution if a choice (XOR-split or OR-split) was encountered, one representing each option. It will return the empty set if no positive token can move and an end state has not yet been reached. In two of the transformations the token has to be updated.

16. propagatePositiveTokens (EPC, Solution)
17. allSolutions ← \{Solution\}
18. Results ← \emptyset
19. while allSolutions ≠ empty do
20. \( S \leftarrow allSolutions \cdot 0 \)
21. currentMarking = \( S \cdot getMarking() \)
22. allSolutions ← allSolutions/\{S\}
23. \( Q \leftarrow S \cdot getAllTokens() \)
24. while \( Q \neq empty \) do
25. \( \text{token} \leftarrow Q \cdot pop() \)
26. \( a = \text{token} \cdot getArc() \)
27. \( \text{element} = a_2 \)
28. if element.type = function then
29. \( f \leftarrow element \cdot function \_id \)
30. \( l \leftarrow S \_L \cdot 0 \)
31. if \( f = l_f \) then
32. \( S \_seta(a, 0) \)
33. \( S \_setk(a, \text{dead}) \)
34. \( S \_L \leftarrow S \_L/l \)
35. \( S \_serviceTime \leftarrow Solution \_serviceTime + l_{te} - l_{tb} \)
36. \( S \_queueTime \leftarrow Solution \_queueTime + l_{tb} - \text{token} \_time \)
37. \( S \_runTime \leftarrow Solution \_runTime + l_{te} - \text{token} \_time \)
38. \( S \_minimalLeadTime \leftarrow S \_minimalLeadTime + l_{te} - l_{tb} \)
39. \( S \_endTime \leftarrow l_{te} \)
40. if \( l_u \neq \text{token} \_user \) then
41. \( S \_realHandovers \leftarrow Solution \_realHandovers + 1 \)
42. \( \text{token} \_user \leftarrow l_u \)
43. fi
44. if \( \text{token} \_roles \cap \rho(f) = \emptyset \) then

83
45. \textit{Solution.\_requiredHand\_Overs} \leftarrow \\
\textit{Solution.\_requiredHandovers} + 1

46. \textbf{fi}

47. \textbf{if } \bot (\textit{token\_roles}) \cap \bot \rho (f) = \emptyset \textbf{ then}

48. \textit{Solution.\_practicalHand\_Overs} \leftarrow \\
\textit{Solution.\_practicalHandovers} + 1

49. \textbf{fi}

50. \textit{token\_roles} \leftarrow \rho (f)

51. \textit{token\_time} \leftarrow \textit{l}_{te}

52. \textit{Q.\_add\_token}

53. \textbf{fi}

54. \textbf{if } \textit{element\_type} = \textit{event} \textbf{ then}

55. \textit{S.\_set\_sigma}\{a\},0 \textbf{ end}

56. \textit{S.\_set\_k}\{a\},\textit{dead} \textbf{ end}

57. \textit{A} \leftarrow \textit{element\_out\_arcs()} \textbf{ end}

58. \textit{Solution.\_set\_sigma}(\textit{B}, \textit{token}) \textbf{ end}

59. \textit{Q.\_add\_token}

60. \textbf{fi}

61. \textbf{if } \textit{element\_type} = \textit{and-split} \textbf{ then}

62. \textit{S.\_set\_sigma}\{a\},0 \textbf{ end}

63. \textit{S.\_set\_k}\{a\},\textit{dead} \textbf{ end}

64. \textit{A} \leftarrow \textit{element\_out\_arcs()} \textbf{ end}

65. \textbf{foreach } (a \in \textit{A}_{\textit{out}}) \textbf{ do}

66. \textit{newToken} \leftarrow \textit{token.\_clone()} \textbf{ end}

67. \textit{Solution.\_set\_sigma}(\textit{a}, \textit{newToken}) \textbf{ end}

68. \textit{Q.\_add\_newToken}

69. \textbf{od}

70. \textbf{fi}

71. \textbf{if } \textit{element\_type} = \textit{and-join} \textbf{ then}

72. \textit{A} \leftarrow \textit{element\_in\_arcs()} \textbf{ end}

73. \textbf{if } (\forall (b \in \textit{A}_{\textit{in}}: \textit{Solution.\_sigma}(b) = \textit{token})) \textbf{ then}

74. \textit{B} = \{\textit{token t}|\textit{t} \in \textit{Solution.\_get\_tokens()} \land \textit{t.\_arc} \in \textit{A}_{\textit{in}}\} \textbf{ end}

75. \textit{Q} \leftarrow \textit{Q} \backslash \textit{B} \textbf{ end}

76. \textit{newToken} \leftarrow \textit{Merge\_Tokens}(\textit{B}) \textbf{ end}

77. \textit{S.\_set\_sigma}(\textit{A}_{\textit{in}},0) \textbf{ end}

78. \textit{S.\_set\_k}(\textit{A}_{\textit{in}},\textit{dead}) \textbf{ end}

79. \textit{Q.\_add\_newToken} \textbf{ end}

80. \textit{S.\_set\_sigma}(\textit{element\_out\_arcs()}, \textit{newToken}) \textbf{ end}

81. \textit{S.\_set\_k}(\textit{element\_out\_arcs()},\textit{wait}) \textbf{ end}

82. \textit{S.\_total\_wait\_time} \leftarrow \\
\textit{S.\_total\_wait\_time} + \sum_{\textit{t} \in \textit{B}} \textit{newToken\_time} - \textit{t\_time}

83. \textbf{fi}
\begin{verbatim}
85.   fi
86.   if element.type = xor-join →
87.     S.setσ(\{a\},0)
88.     S.setκ(\{a\}, dead)
89.     A_{out} = element.outArcs()
90.     S.setσ(\{A_{out} \cdot 0\}, token)
91.     Q.add(token)
92.   fi
93.   if element.type = xor-split →
94.     S.setκ(\{a\}, dead)
95.     S.setσ(\{a\}, 0)
96.     A_{out} = element.outArcs()
97.     b ← one element of B
98.     C ← B/\{b\}
99.   foreach(\{c\} ∈ C) do
100.     S1 = solution.clone()
101.     S1.setκ(\{c\}, wait)
102.     S1.setκ(B\{c\}, dead)
103.     S1.setσ(\{c\}, token)
104.     AllSolutions.add(S1)
105.   od
106.   S.setκ(\{c\}, dead)
107.   S.setκ(\{b\}, wait)
108.   S.setσ(\{b\}, token)
109.   Q.add(token)
110.   fi
111.   if element.type = or-split then
112.     S.setκ(\{a\}, dead)
113.     S.setσ(\{a\}, 0)
114.     A_{out} = element.outArcs()
115.   foreach(\{X ∈ (\text{Powerset}(B)\setminus\emptyset)\}) do
116.     S1 = S.clone()
117.     S1.setκ(\{X\}, wait)
118.     S1.setκ(A_{out}\{X\}, dead)
119.     S1.setσ(\{X\}, token)
120.     S1.setσ(A_{out}\{X\}, -1)
121.     allSolutions ← allSolutions ∪ S1
122.   od
123.   fi
124.   if element.type = or-join then
125.     A_{in} ← element.inArcs()
126.     if (∀b ∈ A_{in}: S.κ(b) = dead ∨ S.σ(b) ≠ 0) then
\end{verbatim}
\[ B = \{ \text{token } t | t \in \text{Solution.getTokens()} \land \text{t.arc } \in A_{in} \} \]
\[ Q \leftarrow Q \setminus B \]
\[ \text{newToken } \leftarrow \text{MergeTokens}(B) \]
\[ S.set_\sigma(A_{in}, 0) \]
\[ S.set_\kappa(A_{in}, \text{dead}) \]
\[ Q.add(\text{newToken}) \]
\[ S.set_\sigma(\text{element.outArcs()}, \text{newToken}) \]
\[ \text{fi} \]
\[ \text{od} \]
\[ \text{if } S.\text{marking } \neq \text{currentMarking} \text{ then} \]
\[ \text{Results } \leftarrow \text{Results } \cup \{ S \} \]
\[ \text{fi} \]
\[ \text{od} \]
\[ \text{return Results} \]
9.2 Score figures

Figure 46: Calculation of the cost score
Figure 47: Calculation of the flexibility score
Figure 48: Calculation of the quality score
Figure 49: Calculation of the time score