Chain monitoring for Web Services

Thesis paper
Software Engineering Research Group
Chain monitoring for Web Services

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

By

Bobby Bregman
born in Rotterdam, the Netherlands
Abstract
This Master Thesis describes an implementation and realization of Chain monitoring for Web Services for Fortis Insurances Netherlands. Through a detailed definition of the assignment and environment the problem definition is described in detail. This forms a basis to describe and make an inventory of the factors that are of importance for the approach to be taken to implement and realize a solution for the problem definition. The iterative approach cuts the implementation and realization into the iterations event-generation, event-collection and event-monitoring.
Each of these iterations is completed / realized through the definition of an approach based upon the project management method Prince2. Requirements engineering and management is applied in the realization of the Literature study, Functional Design and Technical Design. The focus during the realization lies on the demands and wishes of the target groups, the way of implementing and realizing concepts found in the literature study and the implementation of a generic, feasible solution.

The concepts that have been implemented and realized for the generation and collection of health information about chains of Web Services are Aspect Oriented Programming, Java Management eXtensions and Windows Management Instrumentation. These concepts have been realized in a way that both existing and new .NET and J2EE Web Services realized by the ICT Competence Center J2NET can be easily fitted with event-generation.
Java Management eXtensions and Windows Management Instrumentation are also used for the implementation and realization of the central storage of this health information within the existing infrastructure.
Through a number of different web-enabled views this health information is made available for the monitoring of chains of Web Services. The actual chain monitoring is implemented and realized through the application of the concept Web Services Composition. The Web Services Composition Language BPEL4WS is used to model a chain of Web Services. Via a BPEL-interpreter and a rule engine a chain of Web Services can be monitored real-time. Besides this so called FAB View a view for the monitoring of individual Web Services the ICT View exists.

Application of above described approach and concepts onto the given problem definition has delivered a good end result within Fortis Insurances Netherlands. The evaluation of this graduation project is positive, validation and verification of the iterations and project are mainly positive. The used approach and chosen concepts are not specifically pointed on this problem or the Fortis Insurances Netherlands organization or infrastructure. This approach and used concepts can be easily applied upon other problems, organizations or infrastructures. The used approach can be applied upon a broader scale, the term chain monitoring can be placed in a larger context, Web Services are part of a larger environment of applications.
The deliverables of this graduation project are connected to the starting point of this project, the problem definition. The real application onto the organization and infrastructure of Fortis Insurances Netherlands takes place in the Functional and Technical Design; parts of the solution are implemented and realized there. The deliverables of this graduation project are Fortis Insurances Netherlands specific solutions, the concepts chosen and elaborated are suited for the application of chain monitoring for Web Services, the approach applied and described in this graduation project has proven itself in this graduation project but can be applied on a much broader scale.
Acknowledgements

I would like to thank E.J. Elich, dr. K. van der Meer and Prof. Dr. A. van Deursen for their direction, guidance and assistance. E.J. Elich’s approach, patience and overview were of great value to me during this project, I have truly learned a lot about an ICT organization, innovation and gadgets. I have had a lot of sharp, good conversations and discussions (about a variety of subjects) with dr. K. van der Meer, especially his guidance and experience have helped me give shape to my graduation project.

I wish to thank all my colleagues at Fortis VNL ICT CC J2NET. Especially ir. D.G. Doyle and S. Karmous for their help, good advices and thinking along with me. I want to thank all of my colleagues for showing great interest in my research, sharing their vision and providing me with a lot of information about the Fortis organization.

I would like to thank my two reviewers, G.M. van den Berg and J.L.H.M. Maas for the time they spent reviewing my thesis paper, the effort they made for correcting my not always well formed English text.

I want to thank my parents for giving me the opportunity to study Technical Informatics at the Delft Technical University and for the support they gave me while doing this graduation project. And I would definitely like to thank my brother for his sarcastic jokes, never taking me serious and his time to play games with me online so I could lose my aggression.

And last but definitely not least my girlfriend, I would like to thank her for being there for me and listening to me when I talked in my sleep.
# Table of Contents

## Part 1: Introduction and Preparation

1. Introduction .......................................................................................................................... 1
   1.1 Assignment .................................................................................................................... 2
   1.2 Definitions ................................................................................................................... 2
   1.3 Organization ................................................................................................................. 3
   1.4 Infrastructure and architecture .................................................................................. 5
   1.5 Problem definition ....................................................................................................... 7

2. Global approach ................................................................................................................... 8
   2.1 Reason .......................................................................................................................... 8
   2.2 Target group ............................................................................................................... 9
   2.3 Scope .......................................................................................................................... 9
   2.4 Global demands ......................................................................................................... 10
   2.5 Global approach ....................................................................................................... 11
   2.6 Thesis paper overview ............................................................................................ 12

## Part 2: Event-generation and Event-collection

3. Event-generation and Event-collection: Preparation ......................................................... 14
   3.1 Approach for the 1st and 2nd iteration ..................................................................... 14
   3.2 Scope .......................................................................................................................... 15
   3.3 Target group ............................................................................................................... 15
   3.4 Demands of this iteration ......................................................................................... 16
   3.5 Projects concerning ‘chain monitoring’ within ICT ................................................ 17

4. Event-generation and Event-collection: Literature study ................................................ 18
   4.1 Signal definition ......................................................................................................... 18
   4.2 Literature study ......................................................................................................... 21
   4.3 Conclusion ................................................................................................................ 34

   5.1 Approach of the Functional Design ......................................................................... 36
   5.2 Choice for AOP tool .................................................................................................. 38
   5.3 Signal functional definition ....................................................................................... 41
   5.4 JMX and WMI .......................................................................................................... 43
   5.5 Extending the functional documentation of Web Services ...................................... 44

   6.1 Approach of the Technical Design .......................................................................... 45
   6.2 Event-generation ....................................................................................................... 46
   6.3 Signal implementation for JMX and WMI ............................................................... 47
   6.4 Event-collection ....................................................................................................... 48

7. Event-generation and Event-collection: Realization ....................................................... 49
   7.1 Experiences ............................................................................................................... 49
   7.2 Evaluation ............................................................................................................... 50

## Part 3: Event-monitoring

8. Event-monitoring: Preparation ......................................................................................... 54
   8.1 Approach for the 3rd iteration .................................................................................. 54
   8.2 Scope .......................................................................................................................... 55
   8.3 Target group ............................................................................................................... 55
   8.4 Demands and wishes ............................................................................................... 56
   8.5 Projects concerning ‘event-monitoring’ within ICT ................................................. 57

9. Event-monitoring: Literature study ................................................................................. 58
   9.1 Preparation ............................................................................................................... 58
   9.2 Literature study ....................................................................................................... 59
   9.3 Conclusion ............................................................................................................... 66

    10.1 Approach of the Functional Design ................................................................. 67
    10.2 Presentation .......................................................................................................... 68
    10.3 Storage .................................................................................................................... 73

11. Event-monitoring: Technical Design ........................................................................ 77
    11.1 Approach of the Technical Design ....................................................................... 77
    11.2 Presentation layer .................................................................................................. 78
    11.3 Storage layer ......................................................................................................... 81
# Event-monitoring: Realization

12.1 Experiences

12.2 Evaluation

---

# Validation / evaluation

13.1 Approach of the validation

13.2 Evaluation results

---

# Conclusions and Future Work

14.1 Conclusion

14.2 Future work

---

# Bibliography

---

# Appendices

Appendix A: List of Terms

Appendix B: List of Figures

Appendix C: List of Tables

Appendix D: List of other documents
Part 1: Introduction and Preparation
1. Introduction

This chapter discusses the introduction of this thesis paper. The introduction consists of the assignment and a description of the context in which the assignment will be executed. The chapter is closed with a problem definition that is the basis of this thesis.

1.1 Assignment

The assignment for this graduation project and this thesis paper is given by Fortis VNL ICT and stated as follows:

Implement chain monitoring for J2EE and Microsoft .NET Web Services.

Monitor the following aspects:
- Performance
- Availability
- Errors

This assignment is very short and describes no contextual information. The assignment and the description of the contextual information that is done in the remainder of this chapter form the basis of the remainder of this thesis paper.

1.2 Definitions

The assignment basically consists of two terms: ‘chain monitoring’ and ‘Web Services’. This paragraph defines these two terms.

1.2.1 Web Services

Two definitions of Web Services are:
- Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains. They can be local, distributed, or Web-based. Web services are built on top of open standards such as TCP/IP, HTTP, Java, HTML, and XML. ... [1]
- Open standard (XML, SOAP, etc.) based Web applications that interact with other web applications for the purpose of exchanging data. Initially used for the exchange of data on large private enterprise networks, web services are evolving to include transactions over the public Internet. [2]

These definitions mention the facts that Web Services are built upon open standards and that they interact with each other through communication over networks.

1.2.2 Chain monitoring

Chain monitoring or the monitoring of chains is a common term with a lot of applications, to determine what chain monitoring means for this assignment the term ‘chain monitoring’ is split in ‘chain’ and ‘monitoring’.

What do those two terms mean with respect to Web Services? Chain monitoring for Web Services is the whole assignment, Web Services cannot be seen apart from chain monitoring.

A ‘chain of Web Services’ can be interpreted as a number of Web Services that are somehow linked together; interaction between those Web Services takes place.

‘Monitoring Web Services’ is the continuous process of surveying the status of Web Services, checking to see if the Web Services meet certain quality / health demands.

Combining the explanation of ‘chain’ and ‘monitoring’, chain monitoring is the continuous process of surveying the status of a number of Web Services that are linked together to see if they meet certain quality / health demands.
1.3 **Organization**
This paragraph discusses the organization in which the assignment takes place. What does the organization look like and what restraints does the organization put on the assignment.

1.3.1 *Fortis Insurance Netherlands*
The organization on which this graduation project consists basically out of 4 parties (also see Figure 1):

- **Fortis Insurances Netherlands (shortly FIN)**
  FIN is the coordinating insurance company of Fortis within the Netherlands.
  This graduation project is being done within the ICT staff of Fortis Insurances Netherlands (shortly ICT). The ICT staff is divided into Competence Centers; each one of these CC’s has its own competence.
  The CC J2NET has the competency of realizing J2EE and .NET Web Services. Within this CC this graduation project will taken place.

- **Fortis ASR**
  FIN coordinates insurance companies like Fortis ASR, ‘De Amersfoortse Verzekeringen’, Ardanta, ‘de Europeesche Verzekeringen’ and Falcon. These insurance companies are not direct writers but sell insurance policies to consumers via intermediaries.
  The department within these insurance companies that is closest to ICT is the department ‘Functional Application Management’ (shortly FAB); interaction from ICT to insurance companies will take place via this department. The department of Functional Application Management is concerned with the operational management of applications Fortis ASR in this case uses.

- **Intermediaries**
  Intermediaries are linked to insurance companies and sell insurance policies to their customers.
  This party is not of direct concern to this graduation project because there is no communication with this side. This side has no demands and no area of concern with the products delivered in this graduation project.

- **Consumers**
  This side is responsible for buying insurances via intermediaries.

\[\text{Parties within this graduation project} \quad \text{Participation per party} \]

\[
\begin{array}{c|c}
\text{Consumers} & \text{Buying insurance policies via services offered by intermediaries} \\
\hline
\text{Intermediaries} & \text{Offering services to consumers via services offered by insurance companies} \\
\hline
\text{De Amersfoortse Verzekeringen, Fortis ASR} & \text{Insurance companies: Offering services to intermediaries} \\
\hline
\text{CFO Office, Communication, ICT, Facility Management, Human Resources, Fortis Insurances Netherlands} & \text{ICT: Developing services that insurance companies like Fortis ASR can offer to intermediaries} \\
\end{array}
\]

\[\text{Figure 1: Organization of ‘Fortis Insurances Netherlands’}.\]
1.3.2 Prince2
PRINCE2 is a project management method that is widely accepted and used within FIN; all the projects done within the ICT staff are setup and managed via PRINCE2. Because the implementation of this graduation project is done within the ICT staff of FIN, the management of this graduation project is also done in the PRINCE2 way.

The global approach that is followed within this graduation project will be described below. Deliverables of this approach are mentioned within the approach and are separately supplied next to this thesis paper.

Global approach of this project:
- Ensuring the information that is needed for this project is available
  Deliverables of this action are:
  o A Project Initiation Document which contains the information needed to start the project.
    (No PID is made for this project because of the limited resources that are working on this project and the fact that it is no officially mentioned project within Fortis VNL-ICT).
  o A Project Proposal which contains the information needed to justify the project.
- Dividing the project into concrete phases
  Deliverables of this action are:
  o A Project Approach in which the division of the project into phases is described.
- Per phase:
  o Making a planning
    Deliverables of this action are:
    ▪ A Phase Plan in which the planning for this phase is described
  o Realizing the phase
    Deliverables of this action are:
    ▪ A Functional Design of the products delivered in this phase
    ▪ A Technical Design of the products delivered in this phase
- Keeping track of the progress of the project.
  This is done in a non-PRINCE2 way. Through a dialogue with the supervisors is decided that the progress will be discussed through meetings every 3 weeks.
  These meetings consist of an hour and the student and both supervisors will discuss the progress of the graduation project.
  The student delivers the content of those meetings and the supervisors will review that content. When problems are discovered or foreseen these will be discussed in the meetings and measures will be taken.
1.4 **Infrastructure and architecture**

This paragraph discusses the infrastructure in which the Web Services operate and the architecture according to which these Web Service are realized. The purpose of this paragraph is to provide an overview of the area in which this assignment will be realized.

1.4.1 **Infrastructure**

The environment in which the Web Services will be hosted and monitored is part of the infrastructure. The environment will be described with the focus on Web Services; Web Services are the key interest in this assignment so only elements of the infrastructure concerning Web Services will be described here.

**Web Services**

Within FIN and more specific within CC J2NET Web Services are implemented in two different techniques, both these techniques have their own operating environment. The techniques are:

- **Microsoft .NET**
  
  These Web Services use:
  
  - Fortis ASR .NET Framework = this is an ICT made Microsoft .NET Framework that contains functionality concerning process control, data control and user interface controls.
  - Log4Net = an extensive logging plug-in that uses hierarchical loggers for Microsoft .NET technology with which the logging of .NET Web Services can be handled and run-time configured.

- **J2EE**
  
  These Web Services use:
  
  - Sofia Framework = this is an ICT made J2EE Framework that contains functionality concerning process control, data control and business services.
  - Log4J = an extensive logging plug-in that uses hierarchical loggers for Java technology with which the logging of in this case J2EE Web Services can be handled and run-time configured.

**Operating Systems**

Different Operating Systems are used within FIN that host Web Services:

- **Windows**
  
  On the Windows OS the Web Services that are made with help of the Microsoft .NET technology and IIS (Internet Information Server). The run-time environment for Web Services is IIS.

- **AIX**
  
  Advanced Interactive eXecutive is an UNIX like OS from IBM on which at FIN a part of the J2EE Web Services are running. The run-time environment of these Web Services is IBM Websphere.

- **OS/390**
  
  OS/390 is an OS of IBM that runs on an IBM mainframe, on the (IBM) mainframes of FIN the other part of the J2EE Web Services are running. The run-time environment of these Web Services is also IBM Websphere.

**Communication between Web Services**

Communication protocols between Web Services are:

- **SOAP**
  
  Short for Simple Object Access Protocol, a lightweight XML-based messaging protocol used to encode the information in Web Service request and response messages before sending them over a network. SOAP messages are independent of any OS or protocol and may be transported using a variety of Internet protocols, including SMTP, MIME, and HTTP. [3]

  Remark: The messages are being exchanged synchronously between the different Web Services and the emphasis lies on the performance, the message traffic must take place quickly.
1.4.2 Architecture
An important aspect of ICT is the architecture; the most interesting architecture that is used within FIN with regards to this graduation project is the Service Oriented Fortis Insurance Architecture (shortly SOFIA).

Service Oriented Fortis Insurance Architecture
Architecture defines how to use the infrastructure, and contains rules that need to be followed while realizing a piece of functionality and gives guidelines to follow when using a certain environment. The influences of this architecture on this graduation project are now discussed. In short SOFIA defines rules and guidelines of how to use the existing infrastructure and how to realize pieces of functionality of business processes into Web Services. Beneath the most interesting terms with regard to chain monitoring for Web Services will be discussed (see also Figure 2).

Service Oriented Architecture
SOA is an architecture that enables the creation of applications that are built by combining loosely coupled and interoperable services. These services inter-operate based on a formal definition (or contract, e.g., WSDL) which is independent of the underlying platform and programming language. The interface definition hides the implementation of the language-specific service. SOA-compliant systems can therefore be independent of development technologies and platforms (such as Java, .NET etc). [4] A SOA takes care of the fact that functionality isn’t realized twice, when implementing basic elementary functionality in small applications that can be called over a network that functionality can be re-used in different business processes. Business processes consist of multiply pieces of elementary functionality; those pieces of functionality are realized by one or more Web Services. Web Services that operate in a business process are linked together and interact with each other; chains of Web Services are formed.
The difference between Front Office and Back Office
In SOFIA a clear distinction between Front Office and Back Office is made. This distinction is partly responsible for the forming of chains of Web Services and has influences on the choice of technology used for pieces of software realized.

The distinction between Front Office and Back Office is:
- The Front Office consists of applications that are pointed towards interaction with customers like intermediaries or end consumers. The functionality offered in those applications is user-oriented and user interaction is the main focus during the design and implementation of these applications. The technology commonly used for this part of the architecture is Microsoft .NET technology. The infrastructure at FIN is specially designed to easily host Microsoft .NET applications in the Front Office.
- The Back Office consists of applications that are pointed towards internal administration. Functionality offered in these applications is mainly product and information oriented. The technology commonly used for this part of the architecture is J2EE technology. The infrastructure at FIN is specially designed to easily host J2EE applications in the Back Office.

Web Services can consist in both the Front Office and Back Office, which means that .NET Web Services will most likely be part of the Front Office and J2EE Web Services will most likely be part of the Back Office.

1.5 Problem definition
The purpose of this paragraph is to provide a detailed problem definition based upon the preceding paragraphs. The problem definition forms the basis of this thesis paper.

Chain monitoring for Web Services is the initial problem definition, after some research that can be found in the preceding paragraphs the following detailed problem definition can be formed.

Realize chain monitoring for .NET and J2EE Web Services, with regards to the following detailed problems:
- The SOA architecture (SOFIA) divides the infrastructure in multiple environments, through this division environment borders exist.
- Through the SOA architecture and the services definition of SOA, chains of Web Services are needed to offer Back Office functionality to Front Office applications.
- The development of Web Services has lead to a number of Web Services already in production; a lot of Web Services still have to be made. Both existing and new Web Services must be easily fit able with chain monitoring.
- ICT delivers Web Services to customers of ICT, that customers share interests in the monitoring of Web Services, find out what these customers are and what there interests are.

This is the problem definition for this thesis paper; the following chapters form an approach and solution to this problem.
2. Global approach

After the assignment and the context of the assignment described in the preceding chapter this chapter describes the factors that have influenced the chosen global approach of this graduation project. After describing those factors the chosen global approach is discussed.

2.1 Reason

Through a short analysis of the current situation and the future situation a change in situation and the forthcoming need and reason for chain monitoring for Web Services is established.

2.1.1 Current situation

The current situation concerning Web Services that ICT offers contains a number of relatively small Web Services that are managed and maintained by ICT. These Web Services are all being monitored for exception and debug signals by logging statements that are included in the source code of the Web Services. These logging statements can be configured at run-time by the hierarchical plug-ins Log4J and Log4Net (for further details see Chapter 1.4.1 Web Services), these plug-ins make sure that all the information concerning exceptions and debug signals is obtained in log files.

2.1.2 Future situation

The future situation will be that ICT offers a large variety of Web Services, some for internal use, some for external use by intermediaries and end consumers. A lot of the functionality that is implemented in applications other than Web Services will be re-implemented in Web Services. The SOA will be more and more utilized because old functionality will be migrated into Web Services. And the SOA will be more utilized because new functionality will be realized into Web Services. For example offering Back Office functionality to intermediaries will be done through Web Services that can be called from external parties.

All these Web Services are part of small and larger chains of Web Services. And these Web Services offer business critical functionality to a large number of users.

2.1.3 Reason for this graduation project

The difference between the current situation and the future situation is that the number of Web Services offered by ICT to their customers will increase rapidly. Another difference is the offering of Web Services to external parties, calling Web Services from within FIN or from the internet is a whole different world. External parties have higher demands regarding performance, security, availability and errors, these demands require more monitoring and real-time insight in the health of those Web Services.

Currently there is no real-time insight in the performance, availability and errors of all these Web Services and chains of Web Services. The insight in these health issues of chains of Web Services is however critical to make sure that both internal and external users can rely on the functionality these Web Services offer.

Next to the need for ICT to have insight in the health of Web Services they offer to their customers, there is a need to guarantee a certain performance, availability or number of errors. To guarantee those aspects, you first need to be able to measure and monitor these aspects. These two reasons are in short the reasons for this graduation project and thus the need for chain monitoring of Web Services.
2.2 **Target group**

Insight in the target groups and the different domains these groups interact in of this graduation project. A target group is a group of people that have an interest in the problem and solution stated in this thesis paper because they are affected by the problem or solution.

2.2.1 **Target groups**

The target groups that concern the application of chain monitoring for Web Services are:

**ICT**

This party is involved with chain monitoring for Web Services because they realize and maintain the Web Services that are used by Fortis ASR. ICT is responsible for the health of these Web Services that are used by their customers. Fortis ASR is a customer of ICT, but when Fortis ASR offers intermediaries the possibility to use functionality of their Back Office, the intermediaries more or less become customers of ICT. ICT is thus concerned with the health of Web Services because they maintain these Web Services for their customers.

Besides that ICT is first of all concerned with the health of the independent Web Services because ICT realizes independent Web Services, the independent Web Services are deliverables of ICT to the customer. ICT has responsibility for single Web Services and also for chains of Web Services; a chain of Web Services is however of less importance for this party because Web Services are realized and delivered to customers individually and the health of a chain of Web Services comes down to the health of each of the Web Services contained in this chain. The focus thus lies on individual Web Services.

**FAB**

Fortis ASR (also called business) is in this case represented by FAB or Functional Application Management. This party is concerned with chain monitoring because they represent Fortis ASR and the expectations that Fortis ASR has of the operational state of the Web Services. They try to translate these expectations and wishes in statements that ICT can realize and maintain.

This party is interested in the achievements of the whole chain of Web Services because they are responsible for the whole chain or business process to the intermediaries. Thus they carry responsibility for the whole chain of Web Services towards the customers (intermediaries and consumers).

2.2.2 **Domains**

Chain monitoring for Web Services concerns a few parties, chain monitoring can however be applied to two different domains within FIN. Per domain the parties and the interests of these parties differ.

Within FIN there are 2 domains that are important for this project, these domains are:

- **BUSINESS OTA-domain**
  
  In the OTA (develop, test and acceptance)-domain developing software by ICT takes place. This software is being tested by ICT and tested by the customer (Fortis ASR in this case). After these developing-stages the software moves on to the next domain, the BUSINESS P-domain.

- **BUSINESS P-domain**
  
  In this domain the software that is accepted and used by the customer is hosted. This domain is separated from the other domain and is also maintained by ICT.

2.3 **Scope**

This paragraph is concerned with the definition of the scope for this graduation project or defining the content and application area of the project.

The scope of this graduation project is defined through an interpretation of chain monitoring that is used within this graduation project.

FAB and ICT both have different interests in the realization of chain monitoring for Web Services, in this case a chain exists of a business process, and this process falls apart in a number of Web Services. To make chain monitoring possible for both parties the following definition of chain monitoring is used within this graduation project.
Chain monitoring is the monitoring of individual Web Services. Looking at the health of each individual Web Service is important; on top of that the functioning of a chain of Web Services is important. Saying something about the health of a chain of Web Services means that the individual Web Services are healthy. The processes / transactions that such a chain performs are not of direct interest because the focus at first lies with the individual Web Services. The health of the Web Services can be subdivided into performance, availability and errors of Web Services.

2.4 Global demands

After discussing the reason for the application of chain monitoring for Web Services and the involved target groups, the global demands of these target groups are now discussed in this paragraph.

Global demands and starting points / critical factors

The global demands are the demands of the target groups that are stated at the start of this graduation project. These demands form the starting points of this graduation project. These demands are critical factors for this graduation project because the success of this graduation project is depending on the fulfillment of those factors.

2.4.1 Achieving starting points / critical factors

The critical factors and starting points of this graduation project are determined through:

- Conversations within ICT, conversations with architects, supervisors and developers. The intention of these conversations was: what is of importance to chain monitoring of Web Services. [5]
- Translation of the reason and scope (see Chapter 2.1, 2.2 and 2.3) in concrete demands / wishes.
- Conversations with FAB, the intention of these conversations was: what does FAB expect of chain monitoring, to whom do they carry which responsibility. [6]

2.4.2 Starting points / critical factors

The starting points and critical factors are divided into common demands that are not specifically pointed at the solution of this graduation project. And more specific run-time conditions that are pointed towards the run-time aspect of the solution of this graduation project. The list of critical factors / starting points for chain monitoring for Web Services contains the following points:

Common:

- Existing architecture and infrastructure of ICT hardware and software can not be essentially adapted.
- Chain monitoring can be easily applied to both existing and new Web Services.
- Implementation for both J2EE and .NET Web Services cannot be drastically different.
- Costumer information can not be dealt with in the monitoring data.
- Information concerning health of Web Services and chains of Web Services can be saved. Summaries and overview of that information that is not real-time information (but in the past) can be made.

Run-time conditions:

- The performance of Web Services needs to be affected as little as possible, because of the demands that are set for availability, performance and scalability and also the load of the Web Services towards the amount of customers using the Web Services.
- Chain monitoring of Web Services can take place real-time.
2.5 Global approach

The purpose of this paragraph is to provide a detailed approach for this thesis paper based upon the preceding paragraphs. The approach forms the basis for the following chapters of this thesis paper.

2.5.1 Important factors

Factors that have influenced the chosen approach can be found in preceding paragraphs and chapters. These factors all play a role in what approach to choose, these factors are:

- Chain monitoring is a broad term, the exact definition of chain monitoring differs per target group.
- An existing architecture and infrastructure is in place and cannot be changed dramatically.

Other factors that play a role are the factors related to the demands and wishes of ICT:

- Implementation of chain monitoring for both .NET and J2EE Web Services needs to be generic and thus an identical solution for both techniques, the implementation will be driven by the existing infrastructure but the idea can be easily carried out to other infrastructures.
- The SOA architecture leads to the fact that one Web Service can take part in multiple chains of Web Services. A generic and easily adaptable solution for the dealing with chains of Web Services needs to be found.

2.5.2 Division into iterations

The chosen approach is divided into iterations, the reasons for this division are:

- Chain monitoring is a broad term, different parties have different interests which has lead to a complicated problem definition. To better manage and find a solution for this problem definition an approach divided in iterations leads to a broken down problem which is better manageable.
- The existing infrastructure that consists of a number of different environments makes it difficult to find and deliver a single solution, implementation and realization because of the fact that a number of different environments have to be dealt with.

The division into iterations is lead by the definition of chain monitoring stated in Chapter 2.3.

In below mentioned overview the steps will be summed and a translation of steps to iterations will be done. These iterations form the main parts of this graduation project:

- To monitor the health of the individual Web Services information out of the Web Services needs to be generated and collected. Information about the health of Web Services needs to be generated out of the Web Services themselves or the environment surrounding the Web Services. This information also needs to be collected to make monitoring of the health of Web Services possible. This above described step can be subdivided in two iterations:
  o Make Web Services generate events that contains information about the health of those Web Services.
  o Collect those events into a registry or repository.
- To make sure that after those two iterations something can be said about the individual Web Service and the chain of Web Services this information disclosed from the Web Services needs to be translated to visual information about the actual health of the Web Services and chain of Web Services. This all comes to the following iteration:
  o Monitor events through some sort of dashboard-tool at which the health of Web Services and a chain of Web Services can be visualized.

2.5.3 Iterations

The chosen approach thus consists of the iterations event-generation, event-collection and event-monitoring. These iterations are managed by Prince2. Prince2 is the project management method commonly used within ICT; therefore this method applies to this project too. The chosen approach and realization of each iteration will be influenced by this project management method Prince2.
2.6 Thesis paper overview

An overview of what is yet to come in this thesis paper, the approach globally defined the structure of this thesis paper; this paragraph gives a summary of the contents of that structure.

2.6.1 Part 2 and 3: Event-generation, Event-collection and Event-monitoring

Through a description of the approach for each iteration and the translation of the scope and target group of the thesis paper into a scope and target group of each iteration a start is made with that iteration. The demands and wishes are gathered and the necessary preparation for the literature study is done. The literature study is then described; the concepts are examined on a number of points and a concept that best fits the demands and wishes is chosen. The chosen concept is further elaborated in the Functional and Technical Design, demands and wishes are leading the elaboration on the functional and technical level. The realization is then described through a number of Proof of Concepts and an evaluation of each iteration is done at the end of that iteration.

2.6.2 Part 4: Conclusion and Future work

The conclusion is started with an evaluation of the whole graduation project. The realized deliverables are summarized. A conclusion regarding the contribution of this thesis towards the area of chain monitoring for Web Services and regarding the approach and realization of this graduation project is drawn. And at the end some Future work is discussed.

2.6.3 Evaluation theory

An evaluation of the results of each iteration and the whole graduation project is done. This section describes the theory behind this evaluation. The different evaluations are built upon the definitions of verification and validation.

Verification and validation

Verification means that one verifies that a realized product satisfies or matches the original design. [41]
Validation means that one validates that a product design satisfies or fits the intended usage. [40]
The approach or chosen solution and the implementation and realization of this approach are concerned with verification between them. The problem definition and the concepts described in the literature study are concerned with validation between them. For an overview of evaluation divided into verification and validation see Figure 3.

Application onto iterations

At the end of each iteration a verification of the chosen approach and the implementation and realization of that iteration is done. At the end of all the iterations a validation is done of the problem definition and the chosen concepts in the literature study.
These different verifications and validation consist of a chosen approach for the verification or validation and the results of these evaluation parts.

Figure 3: The relationship between validation and verification
Part 2:
Event-generation and Event-collection
3. Event-generation and Event-collection: Preparation

This chapter describes the preparation done for the event-generation and event-collection iterations. The chosen approach, scope and target group is explained. This chapter forms the basis of these iterations.

3.1 Approach for the 1st and 2nd iteration

The 1st and 2nd iterations will be filled in as follows:

- Preparation (see this Chapter):
  - Definition of scope and target group of event-generation.
    The scope and target groups of the global approach are translated to scope and target groups of the current iteration.
  - Description of the demands for the iteration of event-generation.
    The demands, starting points and critical factors that are known and exist with regards to a possible realization of event-generation are stated here.
  - Projects concerning chain monitoring and event-generation within Fortis ASR
    A summary of all projects within ICT that can concern chain monitoring or event-generation.

- Literature study (see Chapter 4):
  - Signal specification
    What do the signals look like that are going to be generated from the Web Services and represent the health of a Web Service and the heart of the chain monitoring solution.
  - Literature study and results of this literature study
    Literature study to event-generation and event-collection plus the results of this study.
  - Conclusion
    Conclusion of this literature study.

- Function Design (see Chapter 5):
  In this phase the following steps will take place:
    - The demands that are formulated within the literature study are translated to demands for the functional design, conversations with ICT an FAB concerning those demands are described.
    - The concepts that will form the basis of event-generation and event-collection derived from the literature study are examined and are further elaborated in the Functional Design; this elaboration uses the demands formulated at the beginning of the Functional Design.
    - After elaboration of the concepts on functional level, gearing these concepts with ICT architects will take place (FAB does only play a part in the event-monitoring iteration because that is where their demands lay, elaboration at this level will not be of influence to their demands). With these architects validation of these concepts will take place to see if the demands that were formulated at the beginning of the Functional Design are met.

- Technical Design (see Chapter 6):
  - The demands that are formulated within the literature study are translated to demands for the technical design, conversations with ICT an FAB concerning those demands are described.
  - The concepts that will form the basis of event-generation and event-collection derived from the Functional Design are examined and are further elaborated in the Technical Design; this elaboration uses the demands formulated at the beginning of the Technical Design.
  - After elaboration of the concepts on Technical level, gearing these concepts with ICT architects will take place. With these architects validation of these concepts will take place to see if the demands that were formulated at the beginning of the Technical Design are met.

- Realization of the Technical Design and delivery of documentation concerning the realized products will take place.

Remark:
The preparation of the 1st and 2nd iteration is mainly pointed towards the 1st iteration, the link with the 2nd iteration becomes clear during the literature study, the end of the literature study and the Functional and Technical Design are pointed towards both iterations.
The above approach is abstracted out of the following items:

- The first and second iteration are described together because of their similar only technical background and because of the linked literature study concepts of these two iterations. The initial approach was not to link these two iterations but during the literature study the link between these iterations became clear. The following paragraphs and the start of the literature study is therefore mainly pointed towards the iteration event-generation, during the literature study the preparation for the iteration event-collection will be described.

- Necessary interpretation of a graduate project within the graduation direction of Software Engineering in the direction of Computer Science at the University of Technology in Delft.

- The choice of standard management method within ICT. This standard management method is Prince2 and contains the formulation of a Functional Design and Technical Design and the delivery of documentation concerning the implemented products.

### 3.2 Scope

The scope of event-generation and event-collection is translated from the global scope stated in Chapter 2, together with the global approach defined in Chapter 2.5 an application onto the iteration of event-generation and event-collection is made here.

#### Scope

The scope is described as the generation and collection of health information of .NET and J2EE Web Services. Health can be subdivided into different areas:

- Performance
- Availability
- Errors

Health information concerns information about the operation of .NET and J2EE Web Services, specific information about the environment is not part of this health information.

The application area of the first and second iteration consists of the .NET and J2EE Web Services made by ICT and the environment these Web Services operate in (see Chapter 1.4.1 for a detailed description of this environment).

### 3.3 Target group

The target groups and their interests in event-generation and event-collection are described here. A translation from target groups and global approach onto the iterations of event-generation and event-collection is made here.

#### ICT

The focus of event-generation and event-collection lies with the Web Services made by ICT. The way performance, availability and error signals are generated and collected is of special interest to ICT because the ICT infrastructure is concerned with the implementation and realization of event-generation and collection.

The information represented by these signals is also of a concern to ICT because it contains the information about individual Web Services that ICT wants to see.

#### FAB

Web Services are not the main concern of FAB; however they have an interest in these two iterations. The information represented by the signals needs to tell something about chains of Web Services eventually. Interests regarding the information represented by these signals will be investigated.
3.4 **Demands of this iteration**

The demands that are linked to the event-generation iteration. These demands, starting points and critical factors are the basis of the literature study to event-generation and the literature results as well as the conclusion of this literature study.

**Demands and starting points / critical factors**

The demands for this iteration are the demands of the target groups that are stated at the start of this graduation project translated towards demands for the target groups of this iteration. These demands form the starting points of this iteration. These demands are critical factors for this iteration because the success of this iteration is depending on the fulfillment of those factors.

3.4.1 **Achieving starting points / critical factors**

The starting points and critical factors are divided into common and run-time demands that are translated from the global demands stated in Chapter 2.4. And more specific signal conditions that are pointed towards the content of the signals that represent health information about the operation of Web Services.

The critical factors and starting points of event-generation are determined through:

**Common / run-time factors:**
- Conversations within ICT, conversations with architects, supervisors and developers. [7,8]
- Translation of the reason and scope (see Chapter 3.2) in concrete demands / wishes.

**Signals:**
- Conversations with FAB, the intention of these conversations was: what does FAB expect of the signals, to whom do they carry which responsibility. [6]
- Conversations with ICT, the intention of these conversations was: what is of importance to monitoring of Web Services. [7,8]

3.4.2 **Starting points / critical factors**

The list of critical factors / starting points for event-generation and monitoring of Web Services contains the following points:

**Common:**
- Existing architecture and infrastructure of ICT hardware and software can not be essentially adapted.
- Event-generation must be applied to both existing and new Web Services.
- Costumer information can not be dealt with in the events.

**Run-time conditions:**
- The performance of Web Services needs to be affected as little as possible, because of the demands that are set for availability, performance and scalability and also the load of the Web Services towards the amount of customers using the Web Services.
- Monitoring of Web Services can take place real-time.
- Information concerning signals can be saved. Summaries and overview of signals that are based on information and signals that are not real-time (in the past) can be made.

**Signals:**
- Performance signals have to tell something about the performance of Web Services and parts of Web Services.
- Availability signals have to tell something about the availability of Web Services and parts of Web Services.
- Fault signals have to tell something about exceptions that occur during the execution of Web Services and those exceptions that fail to fulfill the execution.
- At first it is important that individual Web Services can be monitored on health, eventually something must be said about a whole chain of Web Services and their connection to a business process.
3.4.3 Preview of next iteration
Factors that play a role in the next iteration(s):
- Leave as many possibilities as possible open for the next iteration event-collection.
- Form of the contents of the events must be as flexible as possible so that the demands for event-collection can be met.

3.5 Projects concerning ‘chain monitoring’ within ICT
Within the organization of Fortis VNL and ICT chain monitoring or a closely related form of monitoring has already been realized in several projects, this paragraph is a summary of those projects and the outcomes of those projects, the influence of every project on this graduation project is also discussed.

3.5.1 Project ‘Ketting’
‘Ketting’ was an initiative of ICT to monitor all the systems within Fortis ASR through some sort of radar screen. All systems (mainframes, web servers etc...) needed to define some sort of interface that generated events to a central collection bin, these events would be translated to radar dots on the radar screen, and this translation would be done by a broker. The idea behind this project didn’t make it because the deliverables of this project towards the business (Fortis ASR) only came in the end phase of this project. Two phases of this project were executed; the first phase was a research assignment towards the possibility of coupling all systems within Fortis ASR to a radar screen. The second phase was to implement a part of the event generating interfaces of the systems within Fortis ASR. [9]

Within this project a few sorts of monitoring appear:
- Transaction monitoring
  The monitoring of a transaction that for instance is initiated through a user of an intranet application, the result of this transaction is used as a measure for Service Level Agreements that are made with the customer of those monitored systems.
- Unit monitoring
  Monitor the different BackOffice systems (IBM mainframes for instance, all these systems have their own interface for event generation) and look at system information like CPU usage, memory load.

This project has not been completely realized and the current realization only offers unit monitoring, unit monitoring is not part of this graduation project and will therefore not be taken into account for the conclusion of this report.

3.5.2 Project ‘End-to-end monitoring’
‘End-to-end monitoring’ is a continuation of ‘Ketting’, this project realized transaction monitoring for Cockpit (the extranet application of Fortis ASR). The transaction monitoring is implemented through a Rational Robot, this Rational Robot fires user transactions into Cockpit via the user interface, through measuring of the result screens shown in the user interface the result of the transaction is measured. The results are stored in Tivoli Enterprise Console, a big collection bin that can generate several overviews concerning these results. These overviews serve as an input for Service Level Agreements and as a basis for improving the system. Transaction monitoring is definitely interesting for this graduation project; however the chosen way of realization to measure the health of a system by initiating user actions and measure the health of a system through screens that are shown to the user is not part of the scope of this project. The results are therefore not taken into account for the conclusion of this report. [10]

3.5.3 Project ‘2e Fase RDW’
One of the first Web Services realized by ICT was monitored for health in the following way. The logging plug-in Log4J was used to used to log exceptions and the other events generated from within the Web Service were passed to TEC with the help of SysLog. From TEC different overviews could be generated to see what the health of the Web Service was. The solution to fire events to TEC was not maintained for a long time, TEC was unsuitable for receiving more than one event a second and that was the reason for not maintaining this solution. Since the monitoring of one Web Service didn’t work out with TEC this solution will not be taken into account for this project because this project deals with more than one Web Service. [5]
4. Event-generation and Event-collection: Literature study

This chapter describes the literature study to event-generation and event-collection, the preparation, the results and the conclusion of that literature study. The literature study to event-generation and event-collection is the basis for the functional and technical design of event-generation and event-collection.

4.1 Signal definition

The signals that are to be monitored, the signals are filled in and on this signals the literature study will take place and the literature results will be examined by this signals.

4.1.1 Common Interpretation of signals

The critical factors / starting points (see Chapter 3.4) and the definition of chain monitoring (see Chapter 1.2.2) determine the definition of the signals. The signals form the basis of the literature study.

Background of interpretation of signals

Responsibility

The signals are a concern to FAB because they are responsible for the chain of Web Services. FAB needs to tell something about the chain of Web Services and what its health is now and was in the past. Chain monitoring gives FAB the possibility to gather information about the health of a chain of Web Services, this way FAB can answer complaints and questions of their customers. Justification for the health of individual Web Services is done by ICT, they are after all the developers of these Web Services and they must be able to answer questions of FAB concerning the health and state of the Web Services.

Different levels

A clear division can be made between the chains of Web Services that represent business processes and the individual Web Services that solely form a part of the chain and the business process. The chains of Web Services are important to FAB, they are more or less responsible for this chain to their customers and next to the chain there are individual Web Services that are important for ICT because they are responsible for the health and state of the individual Web Services. Within a Web Service there can be made a further sub division in service methods (when a Web Service is called a service method will be executed) per Web Service. Within a service method a division can eventually be made towards well separated tasks / transactions. This last division only happens when a clear division can be made between tasks / transactions; otherwise a division will only create obscurity about the source of signals.

Important in above sketched picture is that to tell something about the health of a chain of Web Services it is important that something can be said about the health of the individual Web Services and to tell something about the individual Web Services it is important that something can be said about the different service methods and tasks / transactions as shown in Figure 4 (see next page). Generation of signals out of Web Services on a low level can be a way of eventually justifying the health of Web Services on higher levels.
Signals per level
Signals can be subdivided into different levels and can thus be monitored on different levels. Figure 4 shows the different levels and every level is constructed out of the underlying levels. An overview and description of those levels is shown beneath:

- **Process level**
  The process level is the level on which the entire business process can be overseen; from the beginning to the end of the process the health of the process can be monitored. This level is of particular importance to FAB because they carry responsibility towards their customers for this level. The generation and monitoring of signals on this level is important for FAB because their customers can only see this level.

- **Service method level and task / transaction level**
  A business process can be sub divided into a number of calls to service methods of (different) Web Services; every Web Service has one or more service methods that can be called. When a service method is called the level of concern is service method level. At service method level the health of a Web Service at service method level can be monitored. A service method can be sub divided into a number of tasks / transactions. The task / transaction level is a level on which per task / transaction of a service method the health of this service method can be monitored. This level is of importance to ICT because it carries the responsibility of these Web Services and to explain the health of these Web Services it is important to monitor the health of a Web Service on a lowest level so a potential problem with the health of a Web Service can be pointed out as specifically as possible.
4.1.2 Per signal
Signals are generated in the environment of the Web Services; signals can only be generated on service method and task / transaction level because the process level is bigger than a Web Service. It is not desirable to generate signals on process level because then it becomes harder for ICT to monitor the health of individual Web Services and the generation of signals becomes more complex because process information becomes correlated into the signals.

The signals are generated from the environment of the Web Services and are meant to say something about the internal status of a Web Service, interesting areas that are outside the Web Services, like system signals about CPU load and hard disk roam are not part of the scope of this iteration, only information about the operation of the Web Services is part of this iteration and graduation project.

In the overview shown beneath per signal what information is presented and in what way the signal needs to be generated is pointed out:

- Performance signal
  The performance of a Web Service can be measured as the amount of time a service method took for execution. Performance signals have to tell something about the duration of the execution of a service method and the duration of individual task / transactions.
  Performance signals are generated at the moment a service method of task / transaction is executed, at that moment something can be said about that particular execution of a service method or task / transaction.

- Availability signal
  The availability of a service method or task / transaction is the fact that a Web Service is available to be executed correctly. Availability signals also have to say something about the availability of executing a service method and at the moment of executing a service method about the availability of a task / transaction.
  Availability signals are generated at the moment a service method or task / transaction is executed to tell something about the availability of that service method or task / transaction.
  To tell something about the availability of a Web Service a signal can be periodically generated or obtained from the Web Service.

- Fault signal
  Fault signals have to say something about the exceptions that occur during the execution of a service method or task / transaction and if they frustrate the execution of a service method or task / transaction.
  Fault signals are generated at the moment a service method or task / transaction is executed, at that moment information about an exception is available.
4.2 Literature study

The chosen approach for the literature study, how the literature results are obtained and the results are described in such a way that a conclusion about the best suitable concept for this graduation project can be drawn in the next chapter.

4.2.1 Approach

As a result of the previous paragraphs and chapters a literature study has been done to event-generation. The found literature has to fulfill the demands, starting points and critical factors (see Chapter 3.4) and an implementation of the mechanisms mentioned in the literature has to fit within the existing architecture and infrastructure (see Chapter 1.4).

The found literature will be recorded in this report via:

- A short but clear description of the literature and possible implementation in the existing architecture and infrastructure.
- An application of the literature onto the existing architecture and infrastructure and an overview of the possibilities and impossibilities considering the demands, starting points and critical factors stated in Chapter 3.4.
- An evaluation of the literature, consideration of the reliability of the technology, best practices and possibilities of the mentioned solution in the realization phase.

The results of the literature study are shown in the following paragraphs:

4.2.2 Web Service Offerings Language
4.2.3 Aspect Oriented Programming
4.2.4.1 Java Management eXtension
4.2.4.2 Windows Management Instrumentation
4.2.5 Event-collection through JMX and WMI
4.2.2 Web Service Offerings Language (WSOL)

Signal specification - WSOL

WSOL is a language for the formal specification of classes of service, several limitations and management statements for Web Services. A class of service is described as a discrete variation of a full service and Quality of Service that is granted by a Web Service. The syntax of WSOL is according to the XML Schema in a way compatible with the Web Service Description Language 1.1. A WSOL-file points to one or more WSDL-files and adds information that is not available in WSDL-files. [11]

The most important categories of concepts in WSOL are:

- **Service-offering**
  The formal representation of one class of service that contains formal definitions of various constraints and management statements as well as different possibilities for reusable concepts.

- **Constraints**
  Constraints contain Boolean expressions (also arithmetic’s, date / time / duration and some simple series of words) that explain that one or more condition is evaluated.

- **Management statements**
  Every concept, without constraints, that displays important management information about the represented class of service.

- **Reusable concepts**
  Important: service offerings can be defined as extensions of other service offerings.

- **Static and dynamic relationships between service offerings**
  These relationships show similarities and differences between the service offerings and can change runtime (dynamic) or do not change (static) runtime. These relations are specified in a special XML format outside the definitions of the service offerings.

The class diagram of WSOL looks as follows (see Figure 5).

![Partial UML class diagram of the WSOL concepts. [11]](image)

**Event-generation - Implementation**

To monitor the WSOL constraints a management infrastructure is needed, this infrastructure validates the SOAP-messages routed between the Web Services with the help of the WSOL specified constraints. [12]

This management infrastructure consists of the following parts:

- **SOAP-engine**
  SOAP messages must be received, processed and send. [12]

- **WSOL-validation-engine**
  WSOL constraints must be read and applied to SOAP messages and the results of this application must be written to an interface / collection bin. [12]

- **WSDL description of the Web Services**
Examples of an implementation of that management infrastructure are:

- **Web Services Offerings Infrastructure**
  This is a management infrastructure based upon Apache Axis and is extended with a number of extensions to enable the validation of WSOL constraints. [12]

- **“XML Grammar and Parser for the Web Service Offerings Language” by Kruti Patel, B. Eng.**
  This is the MSc Electrical Engineering thesis of a student of the Carleton University in Ottawa, Ontario, Canada. This student has extended an XML parser with a number of extensions that make monitoring of WSOL constraints possible. [13]

**Application within this graduation project – Possibilities and impossibilities**

In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment and reason of this graduation project.

**Possibilities:**

- **Flexible creation of classes of service**
  Creation of classes of service can be done via several concepts (see explanation of WSOL), and relations between are manageable. This offers a great number of possibilities in creating classes of service.

- **Existing and new Web Services do not have to be adjusted**
  Classes of service are not physically tied to Web Services that are monitored via classes of service; classes of service are defined in separate files.

- **Performance of Web Service stays equal in principal**
  The performance of the individual Web Services will not decrease, because the monitoring process takes place on the communication between the Web Services. Problem here however is that the performance of the message traffic (see Chapter 1.4.1 for details) can decrease when the SOAP messages send between Web Services are validated and monitored, depending on the chosen implementation.

**Impossibilities:**

- **Limits of classes of service**
  Despite the number of concepts that can be used to create classes of service there are limitations towards definition of classes of service. Unlimited freedom in defining classes of service is impossible.

- **Scope of WSOL is bigger than just monitoring**
  The target of WSOL is not only monitoring Web Services but also managing Web Services, this graduation project only deals with monitoring, the management part of WSOL will form some sort of overhead because it won’t be used but still exists.

- **Exceptions within the execution of Web Services**
  Exceptions that occur during execution of a service method or task / transaction are not noticed if the results of those exceptions are not visible in the resulting SOAP message.

- **Event-generation on Web Service level**
  Only event-generation on service method level can take place (the Web Service stays some sort of Black Box, the input and output of this Web Service can however be monitored), event-generation and monitoring on lower levels can not take place because there is no possibility in this concept to monitor at lower levels.

- **Availability signals cannot be generated at all**
  The availability of a Web Service and task / transactions cannot be measured, only the availability of a service method can be measured by screening the input and output SOAP messages that come from a Web Service. Availability cannot be seen apart from the execution of a Web Service.
Evaluation
The interception and validation of SOAP messages is not very familiar, looking at the number of applications.
Intercepting and validating SOAP messages is a way of generating events from outside Web Services, this way event-generation has no influence on the execution of Web Services but definitely more influence on the communication between the Web Services. Per SOAP message a big number of aspects have to be abstracted to make event-generation possible, all information with regard to the events has to be abstracted from the message.
Possible attention points for the use of this solution are: limiting the decrease in performance of the communication channels between the Web Services and thus make assumptions about which signals are generated. Standards on how information in those signals needs to be represented will have to be carefully fulfilled and the dependency on those standards will be large.
4.2.3 Aspect Oriented Programming

The idea behind Aspect Oriented Programming (AOP)

AOP is a way of programming a software program based upon the idea that a software program can be better programmed by separating (like in physically separation of source code) specific areas of interests (for instance global exception handling or monitoring) and relying, on the underlying AOP environment, upon the correct merging or weaving of these areas with the source code of a software program. [16]

AOP can be seen as a layer on top of a programming language in which aspects can be specified; those aspects are merged with the source code of a software program by the surrounding AOP environment. Because in AOP specific areas of interest can be specified on a higher level, those areas can be easily controlled and managed apart from the source code of a software program. Take for instance exception handling, in a software program exception handling is applied at several places, when a change in the way exception handling needs to be implemented is made, all those places need to be adjusted, when exception handling is defined as an AOP aspect, those changes only need to be implemented in the AOP aspect. [15]

The aspect consists of source code implemented at specific places in source code and of a specification of places where that aspect needs to be implemented in source code.

Event-generation – Application of AOP

When event-generation is seen as a separate area of interest, events and the generation of events are implemented at several places in the source code of a software program. For instance performance signals, they are implemented at the beginning and end of a service method that is executed, these places and the source code that takes care of the event-generation is specified in an aspect.

When we take a look at the next iteration, the generation of signals can be done in an uniform way and at uniform places in source code when applying aspects to every Web Service that needs to be monitored, the generation of events towards collection bins can take place in the same way in every Web Service that event-generation is applied upon.

Places where event-generation is applied in source code of Web Services are numerous; the way to manage the implementation of event-generation on Web Services through aspects and Aspect Oriented Programming is a good controllable and well-organized and manageable way. When an aspect needs adjustment this has to be done only in this aspect and not in the source code of all the Web Services it is applied on.

The implementation of event-generation through aspects is a well-organized, good controllable and manageable way but has another advantage; event-generation can easily be implemented in existing Web Services.

Latest conditions – J2EE and Microsoft .NET

The latest conditions concerning the implementation of tools that make AOP possible for the technologies J2EE and Microsoft .NET is as follows:

- J2EE [14]

  Implementation of tools:
  A number of different tools, frameworks and add-ins have appeared to implement AOP for J2EE, examples of these implementations are:
  - AspectJ (www.eclipse.org/aspectj)
  - Spring Framework (http://www.springframework.org)

  Status tools:
  From commercially exploitable to concept level.

- Microsoft .NET [14, 17]

  Implementation of tools:
  A number of different tools, frameworks and Visual Studio add-ins have appeared for the implementation of AOP into .NET programming languages, examples of these are:
  - Spring Framework (http://www.springframework.net)
  - Encase AOP Framework (http://theagiledeveloper.com/articles/Encase.aspx)

  Status tools:
  From commercially exploitable until concept.
Application within this graduation project – Possibilities and impossibilities

In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment and reason of this graduation project.

Possibilities:
- Performance signals can be generated
  It is possible to generate performance signals and define the place of generation of these signals in a way that event-generation is possible at service method and task / transaction level.
- Fault signals can be generated
  It is possible to generate fault signals and define the place of generation of these signals in a way that event-generation is possible at service method and task / transaction level.
- Availability signals can be generated
  It is possible to generate availability signals and define the place of generation of these signals in a way that event-generation is possible at service method and task / transaction level.
- Events can be generated at service method and task / transaction level
  Because an aspect offers the ability to specify the place of insertion of a piece of source code event-generation at both service method and task / transaction level is possible.
- Events can be generated at every moment in execution
  Because an aspect offers the ability to specify the place of insertion of a piece of source code event-generation at virtually any moment in execution is possible.
- Implementation of event-generation for new and existing Web Services possible
  Source code of an existing Web Service does not have to be reasonably altered for the implementation of event-generation by means of applying aspects to a Web Service. The aspects are inserted into the source code at in aspect specified places by the AOP environment. The implementation of AOP into a Web Service can be done by altering only a minimal amount of or none source code of the Web Service, this way implementation of event-generation in new and existing Web Services is possible.

Impossibilities:
- Performance of a Web Service will remain the same or decrease a little bit
  The performance of a Web Service will decrease a bit because aspects need to be executed during the execution of a Web Service. The decrease of the performance depends on the chosen implementation of AOP and the contents of the aspects. Next to this performance decrease the implementation of event-generation will decrease the performance of Web Services because current Web Services don’t have event-generation implemented, event-generation will always decrease the performance of a Web Service because the amount of code executed at runtime will always grow with the execution of aspects.

Evaluation
Aspect Oriented Programming is invented a few years ago, some AOP implementations are already commercially exploitable because of the fact that frameworks, tools and books of the AOP implementation have reached a mature phase. AOP implementations are for both Java and .NET available, Java has a slight advantage over .NET considering the maturity of AOP, .NET is however not far away because the implementation of AOP for Java looks a lot like the implementation of AOP for .NET. Applications of AOP are more and more becoming well known and accepted; one of the well known applications of AOP is ‘Transaction Management’, applications that concern monitoring of software programs also become more known. AOP can be used within this graduation project for the implementation of event-generation, the moments in source code of software programs at which events need to be generated and the generation of events can be managed centrally with AOP.

Possible areas of interest are:
- The use of AOP tools, acceptance of these tools within ICT and
- A matching implementation of AOP between J2EE and .NET.
4.2.4.1 Java Management eXtension
Introduction of Java Management eXtension (JMX)
JMX is an aid of Sun Microsystems to make Java resources (applications, services, but also parts of applications) manageable, thus to manage resources.[7] In the case of JMX this means managing a resource until a look at attribute-level can be taken to see what happens with a resource at run-time. [19]
JMX is an integral part of a number of run-time java environments, IBM Websphere the run-time environment of the J2EE Web Services of Fortis ASR is one of them.

JMX architecture
To manage resources an architecture is developed that looks as follows (also see Figure 6):

- Instrumentation level
  This level finds oneself closest to the resources / applications to be managed. At this level so called ‘Managed Beans’ or MBean interfaces are defined, the interfaces are implemented by the resources.
  These interfaces take care of the generation of information from the resources. Different types of interfaces exist: Standard, Dynamic, Open and Model MBeans.
  The interfaces are all suitable for a different type of resource. [18]

- Agent level
  At this level the MBeanServer exists. The MBeanServer consists of a registry that takes care of the handling of the different MBean resources. Next to the MBeanServer there exist different agents on this level that take care of the communication with the next layer. [18]

- Distributed Service level
  At this level the connectors and protocol adaptors exist, these two ways of communication take care of the communication between the management programs and the Agent level. [18]

Event-generation in JMX
To be able to use JMX for event-generation we need to be at Instrumentation level, this level is closest to the monitored resource. By defining an MBean interface that the resource will implement the interface determines what is and what is not monitored. [18]
A resource needs to provide information about the following properties to the MBean interface:

- **Attributes:** Attributes contain the current state of the resource.
- **Constructors:** Constructors are used by a management program to make the resource connect to the MBeanServer.
- **Operations:** Operations are used by a management program to make JMX agents carry out some sort of action on the managed resource.
- **Parameters:** Parameters for constructors and operations
- **Notifications:** Notifications are short messages send from a resource to the MBeanServer, interested listeners can subscribe to a notification through the JMX-notification-infrastructure.

Remark: per type of interface there are different possibilities to implement the information carriers that send information from the resource towards the MBeanServer. For example: a dynamic MBean interface is an interface for which the resource source code is not accessible, a third party software program for instance that does not share its source code. In this case it can be harder to implement the information carriers and to give detailed information about the state of the resource.

**Application within this graduation project – Possibilities and impossibilities**

In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment and reason of this graduation project.

**Possibilities:**

- **Performance signals can be generated**
  By defining an MBean interface and implementing that interface in the resource a performance signal can be send through sending for instance a notification.

- **Fault signals can be generated**
  When certain statements are implemented in source code fault signals can be generated.

- **Availability signals can be generated**
  By defining an MBean interface and implementing that interface in the resource an availability signal can be send through sending for instance a notification.

- **Events can be generated at service method and task / transaction level**
  By implementing the MBean interface in a certain way in the resource differentiation of service method and task / transaction can be made.

- **Events can more or less be generated at every moment in execution**
  By implementing the MBean interface in a certain way in the resource on every moment a signal can be generated.

- **Implementation of event-generation for both new and existing Web Services is possible**
  Remark: Depending on the implemented MBean interface an existing or new resource can implement event-generation. Depending on the MBean interface the information disclosed from the resource will vary in detail and numbers.

- **Monitoring can take place real-time**
  Events can be monitored and generated real-time.

**Impossibilities:**

- **JMX is meant to manage Web Services**
  JMX is not only meant for monitoring Web Services but also for managing Web Services, a part of the functionality of JMX is pointed towards managing Web Services.
  This can both be seen as an advantage as well as a disadvantage.
  When this graduation project is just a start of controlling the health of Web Services managing can be a next step towards full control of Web Services.
  When this graduation project is not a first step towards full control of the health of Web Services a part of the functionality of JMX remains unused and can perhaps cause some overhead.
4.2.4.2 Windows Management Instrumentation

Introduction of Windows Management Instrumentation (WMI)

WMI is a tool from Microsoft to manage applications that run on Windows. WMI is an integrated part of the latest Windows Operating Systems (Windows 2000, XP and 2003) and can be used by applications that run on Windows to manage themselves. An application can be managed by using the WMI interface that is available for several programming languages including Microsoft .NET. The Microsoft .NET technology has a separate interface that is part of the Microsoft .NET Framework. [21]

Architecture of WMI

The architecture of WMI consists of 3 levels [21] (see also Figure 7):

- **Clients**
  This level contains the management applications that receive management information about managed applications and use this information to manage those applications. Information can be obtained from WMI but also operations can be invoked on the management applications and the underlying Operating System.

- **Object manager**
  This is a broker between the provider level and the client level. This layer provides several functionalities like event-publication, event-subscription, event-filtering and query-engine.

- **Providers**
  This level consists of the application to be managed; these applications pass information to the Object manager and carry out operations invoked by the clients through a WMI interface implemented by the application.

Figure 7: Simplified architecture of WMI.
Event-generation in WMI
WMI supports a large event infrastructure that offers various possibilities to publish filter and register events. All of these possibilities can be configured and used by applications to pass information to WMI and give management programs the ability to monitor and manage applications health and status. The interface to use these event-generation possibilities in .NET Web Services is the Microsoft .NET Framework (more specific the System.Management namespace in the .NET Framework). Events can then be generated and published to WMI by implementing certain statements in the source code of a Web Service. [22]

Application within this graduation project – Possibilities and impossibilities
In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment and reason of this graduation project.
Possibilities:
- Performance signals can be generated
  When certain statements are implemented in source code performance signals can be generated.
- Fault signals can be generated
  When certain statements are implemented in source code fault signals can be generated.
- Availability signals can be generated
  When certain statements are implemented in source code availability signals can be generated.
- Events can be generated at service method and task / transaction level
  By implementing certain statements in the source code differentiation of service method and task / transaction can be made.
- Events can more or less be generated at every moment in execution
  By implementing certain statements in the source code on every moment a signal can be generated.
- Monitoring can take place real-time
  Events can be monitored and generated real-time.

Impossibilities:
- Implementation of event-generation on both new and existing Web Services is in principal not possible without making essential modifications to the source code of Web Services. Statements have to be added to the source code to generate events.
- The fact that the performance of Web Services is decreased is not certain, this depends on the implementation of WMI in the Microsoft .NET Framework.
- WMI is not only meant for monitoring but also for managing Web Services, a part of the functionality of WMI is not used but still there. [8]

Evaluation JMX and WMI
JMX and WMI are tested concepts in the area of event-generation, WMI is an integral part of Windows Operating Systems, and JMX is an integral part of IBM Websphere the run-time environment of J2EE Web Services within Fortis ASR.
WMI support is available in the .NET Framework of Microsoft and also available in a number of older programming languages like Visual Basic and VBScript. JMX support is available in a number of Java Frameworks and Sun delivers a number of JMX libraries that are available to use.
Looking at the implementation of event-generation can be concluded that JMX and WMI are tested concepts that considering size and technology have no opponents on their specific platform. WMI has evolved with the new Operating Systems Windows XP and Windows 2003 (Server); JMX has evolved through the many applications in (open source) frameworks.
4.2.5 Event-collection with JMX and WMI

JMX and WMI offer next to event-generation possibilities also some possibilities on the area of event-collection. Because in the case of choosing for the event-generation possibilities the event-collection possibilities cannot be pulled apart this section deals with the possibilities on the area of event-collection. JMX and WMI will be discussed next to other options for the implementation of event-collection.

Critical factors of event-collection

Critical factors with regard to event-collection are:
- Events need to be generated, collected and saved, this in relation to the wish to monitor both past and present.
- The security of the DMZ needs to be taken into account; communication between this environment and the BackOffice is restricted to a number of rules and formats.
- The last iteration event-monitoring needs to be taken into account. Communication between the dashboard tool and collection bins must be made as simple as possible.

Concepts

Looking at the event-collection implementation there are two concepts that can be used to implement event-collection:

1. One event-collection bin per platform or Operating System.
   A decentralized collection bin (see Figure 8). This concept has JMX and WMI as a possible implementation. Per Web Server that hosts Web Services an event-collection bin is available, this collection bin collects all the events that are generated at that Web Server. The communication between event-generation and event-collection is local and has no special restrictions to take care of.

2. One event-collection bin for all platforms.
   A centralized collection bin (see Figure 9). This concept has a possible implementation that uses the hierarchical logging plug-ins Log4J and Log4Net to log all the events generated by Web Services and pass all the events to a central SQL database at a central SQL server. Events need to be communicated to that central server.

Possibilities versus Impossibilities

Per critical factor the balance will be made up in the following overview, JMX and WMI will explicitly be taken into account because that can affect the influence of a critical factor:

- Event-collection
  Both concepts have the ability to collect and store events that are generated by the Web Services. An SQL database can store the events in the database; JMX and WMI rely upon a management application to store events.

- Event-generation, security and event-monitoring
  In the first concept communication between event-generation and event-collection stays local on a Web Server, communication between event-collection and event-monitoring takes place inter-environmental. The communication between event-generation and event-collection can be simple; no need to take security measures into account because this is local communication. Communication between event-collection and event-generation has to take into account the security aspect of inter-environmental communication, JMX and WMI offer various standardized communication possibilities to communicate between event-collection and event-monitoring.
  In the second concept all communication between event-generation and event-collection takes place inter-environmental, a standard communication protocol has to be chosen and dealt with in event-generation, security of this protocol needs to be investigated. Communication between event-collection and event-monitoring can take place locally; security does not have to play a role in this communication.
Figure 8: decentralized collection bin per platform.

Figure 9: centralized collection bin for all platforms.
Evaluation
This section contains an evaluation of the concepts that can be used to implement and realize event-collection. This evaluation is done in 2 parts, first the advantages and disadvantages of a new and existing implementation are described, after that the advantages and disadvantages of both concepts are put together. A summary or balance of these evaluation parts is described in the next section.

If we put the different ways of implementing event-collection together in an overview the following advantages and disadvantages can be assigned:

<table>
<thead>
<tr>
<th></th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>New implementation</td>
<td>- Unlimited choice of event-collection implementation, ways of storing events</td>
<td>- Event collection implementation has to be tested and accepted by the ICT staff.</td>
</tr>
<tr>
<td></td>
<td>and communication to dashboard tool</td>
<td>- Design and implementation has to be done entirely within this graduation project</td>
</tr>
<tr>
<td></td>
<td>- No unused functionality</td>
<td></td>
</tr>
<tr>
<td>Existing implementation</td>
<td>- Solution can be used out of the box</td>
<td>- Unused functionality</td>
</tr>
<tr>
<td></td>
<td>- Concept is already tested, bugs and errors are rare.</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Advantages and disadvantages of a new and existing implementation of event-collection.

This overview shows that an existing implementation offers advantages regarding time and maturity; disadvantage is the possible unused functionality. A new implementation offers advantages considering functionality and choice in implementation possibilities but has disadvantages in the form of time.

When the event-collection concepts are put together, the following overview of advantages and disadvantages is shown:

<table>
<thead>
<tr>
<th></th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Centralized collection bin</td>
<td>- 1 collection bin to connect to for event-monitoring</td>
<td>- Communication between Web Services and collection bin has to commit to security and format standards.</td>
</tr>
<tr>
<td></td>
<td>- 1 data representation, 1 way of collecting data</td>
<td>- Event-generation implementation has to commit to security standards which will increase size of event-generation implementation.</td>
</tr>
<tr>
<td>Decentralized collection bin</td>
<td>- Communication between Web Services and collection bin is local and thus has no restriction in the form of security standards.</td>
<td>- Several (different) event-monitoring implementations to collect monitoring information from.</td>
</tr>
<tr>
<td></td>
<td>- Relative simple event-generation implementation, no restrictions on behalf of security have to be taken into account.</td>
<td>- Several (different) data formats, and ways of collecting data.</td>
</tr>
</tbody>
</table>

Table 2: Advantages and disadvantages of event-collection concepts.

This overview shows that a centralized collection bin offers advantages regarding data representation and complexity of the next iteration; disadvantage is the security and event-generation complexity. A decentralized collection bin offers advantages considering security and event-generation implementation possibilities but has disadvantages in the form of data representations and event-monitoring complexity.
Chain monitoring for Web Services

Summary
Putting the overviews and critical factors together the following balance can be made:

- Event-collection is possible with both concepts
- Generation of signals, passing signals to collection bins and collecting signals for event-monitoring is possible with both concepts.

Requirements for above solution are:

- In the case of a central collection bin a more complicated event-generation implementation with regards to security demands is necessary.
- In the case of a decentralized collection bin a more complicated event-monitoring implementation with regards to security demands is necessary.

4.3 Conclusion
In the preceding paragraph a number of concepts regarding event-generation and event-collection have been described. Every concept is described via a short introduction and an overview of the possibilities and impossibilities as well as an application in the context of this graduation project. This paragraph forms a conclusion to this literature study, a decision is made which concepts are going to be used and implemented for these iterations.

Choice – event-generation
Looking at the concepts for event-generation described in Chapters 4.2.2 until 4.2.5, putting together the possibilities and impossibilities, the following balance can be made.

Possibilities versus Impossibilities
On the basis of the list of possibilities and impossibilities per concept it appears that AOP is most suitable for implementation of event-generation for new and existing Web Services. AOP is suitable for a uniform, manageable way of implementing event-generation. JMX and WMI are tested concepts on the area of implementation of event-generation possibilities, but do not offer the ability to easily, uniformly and manageably implement event-generation for existing Web Services.

Implementation
Combining the event-generation possibilities of JMX, WMI and AOP, AOP can be used to uniformly, easily and manageably implement the event-generation capabilities of JMX and WMI in both new and existing Web Services. This way a tested concept can be implemented for both new and existing Web Services.

Choice – event-collection
Possibilities versus Impossibilities
Both the centralized and decentralized collection offer certain advantages and disadvantages. Considering the fact that JMX and WMI are tested concepts on the area of event-generation and event-collection, offer standardized communication protocols between event-collection and event-monitoring gives JMX and WMI an extra advantage. JMX and WMI are tested concepts that do not give rise to the possibility of implementing a custom implementation of event-generation and event-collection.

Implementation
During the event-collection implementation of JMX and WMI per Web Server aspects like reusability, position with respect to event-monitoring need to be taken into account.
Final choice
The choice that will be made here is both for event-generation and event-collection.
Looking at the implementation of event-generation:
- AOP is a way of implementing event-generation for both new and existing Web Services, AOP is available for both J2EE and .NET technology.
- JMX and WMI are tested concept for implementing event-generation, only offer the ability to easily implement event-generation for new Web Services.
AOP and JMX and WMI offer a tested, good controllable and manageable way of implementing event-generation for both new and existing Web Services. This is the best choice to make looking at the event-generation concepts, demands and wishes.
The choice for JMX and WMI has consequences for the event-collection implementation. JMX and WMI are integrated solutions for both .NET and J2EE event-collection implementation. JMX and WMI offer advantages through standardized ways of communication interfaces and security. The choice for event-collection is therefore made in favor of JMX and WMI.
For an overview of the concepts used to implement and realize event-generation and event-collection see Figure 10.

Figure 10: Event-generation, event-collection en event-monitoring in 1 glance.

The Functional Design of event-generation and event-collection, the approach and the design results of this Functional Design. The Functional Design to event-generation and event-collection is the basis for the Technical Design of event-generation and event-collection.

5.1 Approach of the Functional Design

The approach taken by making the Functional Design, the approach is completed with the demands and wishes that form the basis of the Functional Design.

5.1.1 Approach

The approach that was followed during the making of the Functional Design is as follows:

- Focus on concepts.
  The focus during the making of the Functional Design was on the concepts found in the literature study and the implementation of those concepts that realized the functionality described in the demands and wishes.

- Functionality offered has focus
  The concepts chosen needed to be described in a way that the functionality offered was clear.

- Focus on generic and feasible solutions
  The implementation of concepts also needed to take care of the fact that a generic solution could be made. That solution needed to be feasible and realistic in terms of demands and wishes.

5.1.2 Demands and wishes

A description of the demands and wishes of the functional implementation of event-generation and event-collection through the achievements of those demands and wishes and the actual demands and wishes is made here. Because the focus lies on the concepts chosen in the literature study the demands and wishes are ordered by affected concept.

Achievement demands and wishes

The demands and wishes for the Functional Design of event-generation and event-collection are determined through:

- Conversations with architects within ICT.
- Translation of the demands and wishes of the graduation project and the literature study towards event-generation and event-collection to demands and wishes on a functional level.

Demands and wishes

The list of demands and wishes for event-generation and event-collection contains the following points:

Common:

- Both new and existing Web Services cannot be (drastically) changed by the implementation of event-generation.
- The possibility to switch event-generation on and off at run-time with regard to unforeseen problems is desirable.
- Implementation for both J2EE and .NET Web Services needs to be the same on principles.
AOP:

Remark:
AOP is being used for the first time within ICT through this graduation project. It is of importance that a well described and argued approach is used to implement AOP within ICT. This means a special program of demands is made with input of ICT architects to allow a better acceptance of the technique AOP. This program is shown below:

- Both J2EE and .NET need to have the same AOP implementation structure concerning the clear implementation of AOP.
- Within CC J2NET a number of tools that together form the IDE (see the List of Terms); these tools differ per technology (J2EE and .NET). The different tools are:
  For the technology .NET:
  o Visual Studio 2005
  For the technology J2EE:
  o Rational Application Developer 6
  Implementation of AOP cannot harm the existing IDE, in other words, the current IDE desirably needs be used to implement AOP. The usage of new tools to implement AOP is not going to make the acceptance of AOP and the technique behind AOP better.
- The run-time environment that is currently used to develop and run software with the technologies J2EE and .NET cannot be (drastically) changed to allow the introduction and implementation of AOP.
  The current run-time environments look as follows:
  For the technology .NET:
  o Microsoft Windows 2003 Server
  o Microsoft .NET Framework v2.0
  o Microsoft IIS 6.0
  For the technology J2EE:
  o AIX / OS-390
  o IBM Websphere v5.0
- Performance of a Web Service without AOP and the same Web Service with AOP cannot be drastically less. (in order of percents)
- The concept description of AOP in the literature study states that AOP is a way of abstracting certain aspects of the source-code of a Web Service out of that source-code and make those aspects manageable by placing those aspects in an AOP environment. That description needs to be implemented as well as possible.

Signals:

- Performance, availability and error signals must give information about the execution of the current Web Service on available signal levels.
- Reusability
  Signals can be implemented for every Web Service realized by CC J2NET.

JMX and WMI:

- Signals can be mapped on available event-generation capabilities, signals can be generated during execution of a Web Service.
- Events of all Web Services can be collected
  Event-generation can be implemented for every Web Service realized by the CC J2NET.
- Performance of a Web Service needs to be affected as little as possible by the implementation of event-generation.
  The amount of events generated by the execution of Web Services needs to be kept as low as possible with regards to the inevitable performance loss that occurs while generating events.
## 5.2 Choice for AOP tool

The choice for an AOP tool based upon the factors that describe the functionality of and AOP tool. This paragraph first describes that factors and than the choice on basis of those factors for the technology .NET and J2EE is made.

### 5.2.1 AOP factors

An overview of the factors that are of importance while choosing an AOP tool and a motivation:

<table>
<thead>
<tr>
<th>Factor name</th>
<th>Contents of factor</th>
<th>Motivation of factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source format</td>
<td>The way aspects are saved. Possibilities:</td>
<td>This factor is of importance because of the fact that it can change the IDE. A different source format cannot always be integrated into an existing IDE.</td>
</tr>
<tr>
<td></td>
<td>• Plain java / .net</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Other format</td>
<td></td>
</tr>
<tr>
<td>Run environment</td>
<td>The way aspects are executed run-time. Possibilities:</td>
<td>This factor is of importance because it can affect the IDE and run-time environment.</td>
</tr>
<tr>
<td></td>
<td>• Aspects are in source code</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Aspects are executed by functionality offered in for example a framework.</td>
<td></td>
</tr>
<tr>
<td>Weaving time</td>
<td>The moment that aspects are woven into source code.</td>
<td>This factor is of importance because it can affect the performance of compiling, building and running source code. The performance loss of build-time weaving is zero because aspects are not loaded run-time but are woven in source code. Performance loss of run-time weaving is bigger than zero because run-time source code needs to be enriched by aspects; the amount of loss depends on the implementation of weaving.</td>
</tr>
<tr>
<td></td>
<td>Possibilities:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Build-time = Post-compile time or standard-compile time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Load-time = Standard-compile time when classes are loaded</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Run-time = Interception en proxy-based mechanisms make sure that during execution of a piece of code aspects are executed.</td>
<td></td>
</tr>
<tr>
<td>Compiler requirements</td>
<td>The moment aspects are defined in another language, aspects cannot be compiled by a standard compiler anymore, another compiler needs to be used and implemented instead.</td>
<td>This factor is of importance because a new compiler changes the IDE and a part of the run-time environment, a big change.</td>
</tr>
<tr>
<td>Deployment</td>
<td>Is it possible to turn on and off aspects run-time?</td>
<td>Turning on and off aspects during run-time can fulfill the wish to switch event-generation on and off during run-time. It also makes the aspects more manageable during run-time.</td>
</tr>
<tr>
<td></td>
<td>Possibilities:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ‘hot deployable’ = run-time switching is possible.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• static deployable = no run-time switching</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>What is the idea behind this technique, what is the importance of AOP within this tool?</td>
<td>The use of a technique does not stand on itself, the future (support, additions) of the technique also play a part in the choice of a tool.</td>
</tr>
<tr>
<td>Acceptance within ICT</td>
<td>A short description of the consequences that the use of this AOP tool has within the CC J2NET, possible (dis)advantages of this tool and a description of the consequences of the ‘Program of demands’ stated in the preceding paragraph.</td>
<td>AOP is not being used in the development process of CC J2NET at this moment. A chosen AOP tool must be able to take part in that development process and must be able to fit in that process.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Factor name</th>
<th>Contents of factor</th>
<th>Motivation of factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment</td>
<td>Is it possible to turn on and off aspects run-time?</td>
<td>Turning on and off aspects during run-time can fulfill the wish to switch event-generation on and off during run-time. It also makes the aspects more manageable during run-time.</td>
</tr>
<tr>
<td></td>
<td>Possibilities:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ‘hot deployable’ = run-time switching is possible.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• static deployable = no run-time switching</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>What is the idea behind this technique, what is the importance of AOP within this tool?</td>
<td>The use of a technique does not stand on itself, the future (support, additions) of the technique also play a part in the choice of a tool.</td>
</tr>
<tr>
<td>Acceptance within ICT</td>
<td>A short description of the consequences that the use of this AOP tool has within the CC J2NET, possible (dis)advantages of this tool and a description of the consequences of the ‘Program of demands’ stated in the preceding paragraph.</td>
<td>AOP is not being used in the development process of CC J2NET at this moment. A chosen AOP tool must be able to take part in that development process and must be able to fit in that process.</td>
</tr>
</tbody>
</table>
5.2.2 Choice for the technology J2EE

Comparison

Possible AOP tools for J2EE (per possibility a description of the AOP factors):

<table>
<thead>
<tr>
<th>Factor \ Tool</th>
<th>AspectJ</th>
<th>AspectWerkz</th>
<th>JBoss AOP</th>
<th>Spring AOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source format</td>
<td>Extended .java .aj</td>
<td>Plain .java .xml</td>
<td>Plain .java .xml</td>
<td>Plain .java .xml</td>
</tr>
<tr>
<td>Run environment</td>
<td>Aspects are completely woven into source code. Aspects are completely executed in source code.</td>
<td>Aspects are partly woven into source code. Aspects are executed by a number of mechanisms.</td>
<td>Aspects are partly woven into source code. Aspects are executed by a number of mechanisms.</td>
<td>Aspects are partly woven into source code. Aspects are executed by proxy objects in source code.</td>
</tr>
<tr>
<td>Weaving time</td>
<td>Build-time Load-time</td>
<td>Build-time Load-time</td>
<td>Run-time</td>
<td>Run-time</td>
</tr>
<tr>
<td>Compiler requirements</td>
<td>Incremental AspectJ Compiler needed</td>
<td>Java Compiler and post processing</td>
<td>Java Compiler and post processing</td>
<td>Java Compiler</td>
</tr>
<tr>
<td>Deployment</td>
<td>Static deployment</td>
<td>Hot deployable</td>
<td>Hot deployable</td>
<td>Hot deployable</td>
</tr>
<tr>
<td>Status</td>
<td>Released in 2001, current version 1.2.1. 1 of the first AOP tools on the market. Books about this tool have been published already.</td>
<td>Released in 2002, current version 2.0. Possibly joining with AspectJ.</td>
<td>Released in 2004 as an addition to JBoss application server framework, current version is 1.0.</td>
<td>Released in 2004 as an addition to Spring framework, current version is 1.2.6. Built by modules. A lot of documentation available.</td>
</tr>
</tbody>
</table>

Table 4: Comparison of AOP tools for J2EE.

Choice + motivation

The final choice which AOP tool to use depends on the weight of the factors. Which factors weigh the most and what about the demands issued in the ‘Program of Demands’.

Source Format, Weaving Time, Run environment and Deployment are important factors for developers, the people that are going to implement AOP in their Web Services because these factors influence the way of developing and the IDE. Status and ‘Acceptance within ICT’ are important factors for the ICT organization because AOP needs to earn a place in the development process after this ‘Proof of Concept’ (what this graduation project can be seen like).

When demands and factors are compared to possibilities for J2EE, the following conclusion can be drawn:

- AspectJ is not chosen because the IDE needs to be changed drastically.
- AspectWerkz is not chosen because of the joining with AspectJ, the uncertainty of that has major effects on the acceptance within ICT.
- JBoss AOP is not chosen because the IDE needs to be changed dramatically and less to none experience is available within the ICT organization.
- Spring AOP meets the demands and with the fact that it the Spring Framework is already being used within Fortis it definitely is the best choice that meets all the demands.

The final choice is made to use Spring AOP.
5.2.3 Choice for the technology .NET

Comparison

Possible AOP tools for .NET (per possibility a description of the AOP factors):

<table>
<thead>
<tr>
<th>Factor \ Tool</th>
<th>Encase</th>
<th>Aspect#</th>
<th>RAIL</th>
<th>Spring AOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source format</td>
<td>Plain .net .xml</td>
<td>Plain .net .xml</td>
<td>Plain .net .xml?</td>
<td>Plain .net .xml</td>
</tr>
<tr>
<td>Run environment</td>
<td>Aspects are partly woven into source code. Aspects are executed by proxy objects in source code.</td>
<td>Aspects are partly woven into source code. Aspects are executed by proxy objects in source code.</td>
<td>Aspects are completely woven into source code. Aspects are executed through a mechanism called dynamic loading.</td>
<td>Aspects are partly woven into source code. Aspects are executed by proxy objects in source code.</td>
</tr>
<tr>
<td>Weaving time</td>
<td>Run-time</td>
<td>Run-time</td>
<td>After Build-time At Load-time</td>
<td>Run-time</td>
</tr>
<tr>
<td>Compiler requirements</td>
<td>.NET Compiler</td>
<td>.NET Compiler</td>
<td>.NET Compiler (Special Common Language Runtime manipulate assemblies)</td>
<td>.NET Compiler</td>
</tr>
<tr>
<td>Deployment</td>
<td>Hot deployable</td>
<td>Hot deployable</td>
<td>Hot Deployable Hot Deployable</td>
<td>Hot deployable</td>
</tr>
<tr>
<td>Status</td>
<td>Released in 2005, current version is 0.7. Framework specially build for AOP.</td>
<td>Released in 2004, current version 2.1.1. Framework specially build for AOP.</td>
<td>Released in 2003, current version is 0.5.7. Framework not specially build for AOP.</td>
<td>Released in 2005, current version 1.0.1. Framework is related to J2EE variant. A lot of documentation available.</td>
</tr>
<tr>
<td>Acceptance within ICT</td>
<td>No new IDE. No experience.</td>
<td>No new IDE. No experience.</td>
<td>No new IDE. No experience.</td>
<td>No new IDE. Framework is currently used for J2EE within Fortis.</td>
</tr>
</tbody>
</table>

Table 5: Comparison of AOP tools for .NET.

Choice + motivation

The final choice which AOP tool to use depends on the weight of the factors. Which factors weigh the most and what about the demands issued in the ‘Program of Demands’.

Source Format, Weaving Time, Run environment and Deployment are important factors for developers, the people that are going to implement AOP in their Web Services because these factors influence the way of developing and the IDE. Status and ‘Acceptance within ICT’ are important factors for the ICT organization because AOP needs to earn a place in the development process after this ‘Proof of Concept’ (what this graduation project can be seen like).

When demands and factors are compared to possibilities for J2EE, the following conclusion can be drawn:

- Encase is not chosen because this framework is relatively new and no experience is available with this framework.
- Aspect# is not chosen because there is no experience with this framework, the link with J2EE is difficult because it is an independent / separate framework.
- RAIL is not chosen because it is relatively new and there is no experience with this framework. This framework is also not build especially for AOP, AOP is not the core thus difficulties with acceptance exist.
- Spring AOP is looking at the demands the best choice because there is a strong resemblance with J2EE, a lot of experience on the terrain of J2EE is already available within Fortis. All the demands are met with the usage of this framework.

The final choice is made to use Spring AOP because of the resemblance and the intact IDE.
5.3 Signal functional definition

An overview of the different signals, which information is needed for each signal and where that information can be found is given here. The information described in this paragraph is a summary, a detailed description of the signals can be found in the separately supplied Functional Design document.

In figure 11 the times at which a signal is generated and the levels on which a signal is generated can be seen:

![Figure 11: Signal implementation](image-url)
The overview shown beneath shows the different signals, levels and signal information:

<table>
<thead>
<tr>
<th>Signal</th>
<th>Level</th>
<th>Signal information</th>
</tr>
</thead>
</table>
| Performance | Service method level | Collected information:  
  - Time and date of generation of the signal  
  - Name of the current Web Service and Web Service method  
  - Information regarding the measured length of execution of the service method / task or transaction  
  - Information regarding the correct completeness of this execution |
| Task / Transaction level | Collected information:  
  - Time and date of generation of the signal  
  - Name of the current Web Service and Web Service method  
  - Name of the current task / transaction  
  - Information regarding the measured length of execution of the service method / task or transaction  
  - Information regarding the correct completeness of this execution |
| Availability | Web Service level | Collected information:  
  - Time and date of generation of the signal  
  - Is the Web Service available?  
  - Service method level | Collected information:  
  - Time and date of generation of the signal  
  - Is the Web Service method available?  
  - When not available, if possible, information about the not being available of the Web Service:  
    - What is not available  
    - Why is it not available  
  - Task / Transaction level | Collected information:  
  - Time and date of generation of the signal  
  - Is the Web Service method task available?  
  - When not available, if possible, information about the not being available of the Web Service:  
    - What is not available  
    - Why is it not available  
  - Error | Service method level | An error signal must contain the following information:  
  - Time and date of generation of the signal  
  - Name of the current Web Service and Web Service method  
  - Error message and an error code if available  
  - Task / Transaction level | Because of the fact that errors that block the normal execution of a Web Service can only occur at service method level, error signals are only generated at service method level. |

Table 6: An overview of signals, signal levels and signal information.
5.4 **JMX and WMI**

The functional implementation of JMX and WMI. In the ‘Functional implementation’ section is described how this paragraph is related to the actual Functional Design of JMX and WMI; the other paragraphs show a short summary of the implementation of JMX and WMI.

5.4.1 **Functional implementation**

The functional implementation of JMX and WMI for event-generation and event-collection builds on top of the signal definition stated in the preceding paragraph, that signal definition needs to be implemented in JMX and WMI.

For both the implementation of JMX and WMI in the following sub paragraphs a short summary is given of the actual Functional Design of those literature study concepts, for the whole Functional Design you have to look in the separately supplied Functional Design document.

5.4.2 **Event-generation**

**JMX**

Per Web Service a MBean-interface consists with the following properties:
- Attributes, contain the state of a Web Service.
- Constructors are needed to instantiate the management interface.
- Operations, execute an action on a management interface.
- Parameters of constructors en operations.
- Notifications are send via the JMX infrastructure to interested parties.

The combination of these properties forms the management-interface of a Web Service.

**Implementation possibilities**

Beneath table shows the mapping of signals to management interface properties:

<table>
<thead>
<tr>
<th>Signal</th>
<th>Level</th>
<th>Possibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Service method level</td>
<td>Notifications</td>
</tr>
<tr>
<td></td>
<td>Task / Transaction level</td>
<td>Notifications</td>
</tr>
<tr>
<td>Availability</td>
<td>web service level</td>
<td>Operations = On demand requested</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Via run-time environment -&gt; periodical generation</td>
</tr>
<tr>
<td></td>
<td>Service method level</td>
<td>Notifications</td>
</tr>
<tr>
<td></td>
<td>Task / Transaction level</td>
<td>Notifications</td>
</tr>
<tr>
<td>Error</td>
<td>Service method level</td>
<td>Notifications</td>
</tr>
</tbody>
</table>

Table 7: Signal to management interface mapping for JMX.

Explanation of choices made in the above table: Notifications are suited to realize the event character of signals. Operations offer a possibility to realize availability signals on demand.

**WMI**

Per Web Service a Managed Object Format-class consists with the following properties:
- Managed class that consists of attributes and fields.
- Events are send via the WMI infrastructure to interested parties.

The combination of these properties forms the Managed Object Format-class of a Web Service.

**Implementation possibilities**

Beneath table shows the mapping of signals to management interface properties:

<table>
<thead>
<tr>
<th>Signal</th>
<th>Level</th>
<th>Possibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Service method level</td>
<td>Events</td>
</tr>
<tr>
<td></td>
<td>Task / Transaction level</td>
<td>Events</td>
</tr>
<tr>
<td>Availability</td>
<td>web service level</td>
<td>Periodical generation via run-time environment</td>
</tr>
<tr>
<td></td>
<td>Service method level</td>
<td>Events</td>
</tr>
<tr>
<td></td>
<td>Task / Transaction level</td>
<td>Events</td>
</tr>
<tr>
<td>Error</td>
<td>Service method level</td>
<td>Events</td>
</tr>
</tbody>
</table>

Table 8: Signal to management interface mapping for WMI.

Explanation of choices made in the above table: Events are suited to realize the event character of signals.
5.4.3 Event-collection

**JMX**
Signals are transmitted via notifications to the MBean-server. The MBean-server takes care of the collection of management interface properties.

Other interesting implementation facts are:
- **Data model**
  The way information from management interfaces is stored in the MBean-server is defined by the JMX standard.
- **Event-monitoring**
  The MBean-server offers a number of standardized connectors to communicate between an MBean-server and a dashboard tool.

**WMI**
Signals are transmitted via events to the WMI-registry. The WMI-registry takes care of the collection of management interface properties.

Other interesting implementation facts are:
- **Data model**
  The way information from management interfaces is stored in the WMI-registry is defined by the WMI standard.
- **Event-monitoring**
  The WMI-registry offers a number of standardized connectors to communicate between a WMI-registry and a dashboard tool.

5.5 Extending the functional documentation of Web Services

Description of the functional extension that is made for the already available functional documentation of Web Services within the CC J2NET. First the current situation with regards to documentation is discussed; after that the addition is described.

5.5.1 Current situation
At this moment there exists a so called ‘ServiceView’ document per Web Service within CC J2NET. This ‘ServiceView’ document contains the following information:
- Service characteristics
- Organization and processes
- Functionality
- Demands
- Service Level Agreements
- How to send a request to this Web Service
- Technical characteristics

5.5.2 Addition for the implementation of chain monitoring
The additions that are necessary for the implementation of event-generation and event-collection on a Web Service are:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Category</th>
<th>Place in ServiceView template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event-generation</td>
<td>Common</td>
<td>Functionality \ Functional Description</td>
</tr>
<tr>
<td></td>
<td>Performance</td>
<td>Functionality \ Monitoring-information \ Event-generation \ Performance signals</td>
</tr>
<tr>
<td></td>
<td>Availability</td>
<td>Functionality \ Monitoring-information \ Event-generation \ Availability signals</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>Functionality \ Monitoring-information \ Event-generation \ Error signals</td>
</tr>
<tr>
<td>Event-collection</td>
<td>Namespace / MBean name</td>
<td>Functionality \ Monitoring-information \ Event-collection</td>
</tr>
<tr>
<td></td>
<td>Event-generation</td>
<td>Functionality \ Monitoring-information \ Event-collection</td>
</tr>
</tbody>
</table>

Table 9: Functional documentation extensions needed for the implementation of event-generation and event-collection.

The Technical Design of event-generation and event-collection, the approach and the design results of this Technical Design. The Technical Design to event-generation and event-collection is the basis for the realization phase of event-generation and event-collection.

6.1 **Approach of the Technical Design**

The approach taken by making the Technical Design, the approach is completed with the demands and wishes that form the basis of the Technical Design.

6.1.1 **Approach**

The approach that was followed during the making of the Technical Design is as follows:

- Focus on techniques.
  
  The focus during the making of the Technical Design was on the techniques that realize the concepts described in the Functional Design and the best implementation of those techniques.

- Focus on generic and feasible solutions
  
  The techniques also needed to take care of the fact that a generic solution could be made. That solution needed to be feasible and realistic in terms of demands and wishes.

6.1.2 **Demands and wishes**

A description of the demands and wishes of the technical implementation of event-generation and event-collection through the achievements of those demands and wishes and the actual demands and wishes is made here. The common demands and wishes are described here; the more technique specific demands and wishes are described in the paragraphs describing the implementation of those techniques.

**Achievement demands and wishes**

The demands and wishes for the Technical Design of event-generation and event-collection are determined through:

- Conversations with architects within ICT.
- Translation of the demands and wishes of the Functional Design and from demands and wishes of the graduation project to event-generation and event-collection into demands and wishes on a technical level.

**Demands and wishes**

The list of demands and wishes for event-generation and event-collection contains the following points:

- Both new and existing Web Services cannot be (drastically) changed by implementation of event-generation.
- The possibility to switch event-generation on and off at run-time with regard to unforeseen problems is desirable.
- Implementation for both J2EE and .NET Web Services needs to be the same on principles.
- Performance of a Web Service without AOP and the same Web Service with AOP cannot be drastically less. (in order of percents)
- The concept description of AOP in the literature study states that AOP is a way of abstracting certain aspects of the source-code of a Web Service out of that source-code and make those aspects manageable by placing those aspects in an AOP environment. That description needs to be implemented as well as possible.
6.2 Event-generation

The implementation of event-generation for both J2EE and .NET Web Services. Because the implementation is the same for both technologies the implementation is only discussed once.

6.2.1 Technical implementation

A short summary of the technical implementation of event-generation is discussed. The implementation basically consists out of 3 parts:

- The Web Service itself
  The Web Service itself contains the following implementation details.
  The source code that is responsible for the actual functionality implemented by the service methods and task / transactions of the Web Service is not directly called via the http-request handler but via a proxy factory object, this proxy factory instantiates the needed source code that is contained in a certain source code class. When calling methods (service methods and task / transactions) on the source code class pre defined event-generation aspects are plugged into the execution of the current service method or task / transaction. The proxy factory can be created in source code or via a so called applicationContext document, the applicationContext document is a Spring configuration file in which code classes and their connections are configured.

- The event-generation package
  This package contains the source code of:
  - The aspect oriented programming aspects for generating performance and error signals
  - The connection from event-generation to the event-collection infrastructure

- The event-collection infrastructure
  The infrastructure that makes it possible to collect the events generated by the event-generation software.

6.2.2 Overview

In the enumeration above the individual parts of the event-generation technical implementation are discussed. Figure 12 shows the individual parts and the way they are connected to each other.

Figure 12: Implementation of event-generation
6.3 Signal implementation for JMX and WMI

The implementation of the signals in JMX and WMI event-generation capabilities is discussed; this paragraph is a short summary of the whole technical implementation described in the Technical Design document separately supplied with this thesis paper.

6.3.1 Approach

The approach that was followed during the making of the signal implementation for both JMX and WMI was:

- Describe the run-time environment of the Web Service with regard to JMX or WMI.
- Translate the demands from the Functional and Technical Design into demands for the specific implementation of signals for JMX or WMI.
- The possible implementations and the factors that have influence on the implementation of the signals in JMX and WMI are described. This way a motivated choice can be made to implement the best possible solution with regard to the demands for the implementation of signals.

The demands to the implementation of signals in JMX and WMI are:

- AOP aspects must be pluggable
- AOP aspects can be separated from the source code of a Web Service
- The self defining of aspects / events must be possible
- Quality of the signals must be ensured, a standard package of signal implementations must be made.
- The standard package of signal implementations must be extensible
- The amount of events needs to be kept as low as possible.
- Run-time switching on and off of event-generation must be possible.

6.3.2 JMX implementation

The end result of the signal implementation for JMX. The ModelMBean method is part of ModelMBean management interface, this is a special kind of management interface, the choice for this kind of management interface is (like stated in the paragraph introduction) described in the Technical Design document.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Web Service level</th>
<th>Service method level</th>
<th>Task / Transaction level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Method in ModelMBean</td>
<td>Notification that is part of the performance signal at this level</td>
<td>Notification that is part of the performance signal at this level</td>
</tr>
<tr>
<td>Availability</td>
<td>WebHeartBeatEvent</td>
<td>Event that is part of the performance signal at this level</td>
<td>Event that is part of the performance signal at this level</td>
</tr>
<tr>
<td>Error</td>
<td>Derived from WebBaseEvent</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 10: Signal implementation for JMX.

6.3.3 WMI implementation

The end result of the signal implementation for WMI. The different event names (for instance WebBaseEvent) are special events that are part of the ASP.NET 2.0 Health monitoring feature, the choice for this feature is (like stated in the paragraph introduction) described in the Technical Design document.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Web Service level</th>
<th>Service method level</th>
<th>Task / Transaction level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Derived from WebBaseEvent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Availability</td>
<td>WebHeartBeatEvent</td>
<td>Event that is part of the performance signal at this level</td>
<td>Event that is part of the performance signal at this level</td>
</tr>
<tr>
<td>Error</td>
<td>Derived from WebBaseErrorEvent</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Signal implementation for WMI.
### 6.3.4 Technical signal implementation
The information that is generated for every event, the signal definition that was formed through the implementation of signals for JMX and WMI is described below:

<table>
<thead>
<tr>
<th>Signal Level</th>
<th>Parameters</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service method</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeNaam</td>
<td>Name of the executed service method</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeDurur</td>
<td>Number of execution msec</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeVoltooid</td>
<td>Boolean (completed successfully)</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeDatum</td>
<td>Date + time of generation of event</td>
</tr>
<tr>
<td>Task / Transaction</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeNaam</td>
<td>Name of the executed service method</td>
</tr>
<tr>
<td></td>
<td>TaakNaam</td>
<td>Name of executed task / transaction</td>
</tr>
<tr>
<td></td>
<td>TaakDuur</td>
<td>Number of execution msec</td>
</tr>
<tr>
<td></td>
<td>TaakVoltooid</td>
<td>Boolean (completed successfully)</td>
</tr>
<tr>
<td></td>
<td>TaakDatum</td>
<td>Date + time of generation of event</td>
</tr>
<tr>
<td>Web Service</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td>ServiceDatum</td>
<td>Date + time of generation of event</td>
</tr>
<tr>
<td>Service method</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeNaam</td>
<td>Name of the executed service method</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeFoutBoods chap</td>
<td>Error message</td>
</tr>
<tr>
<td></td>
<td>ServiceMethodeDatum</td>
<td>Date + time of generation of event</td>
</tr>
</tbody>
</table>

Table 2: Technical signal implementation.

### 6.4 Event-collection
The implementation of event-collection in JMX and WMI is discussed; this paragraph is a short summary of the whole technical implementation described in the Technical Design document separately supplied with this thesis paper.

#### 6.4.1 JMX implementation
The end result of the event-collection implementation for JMX. The full implementation is (like stated in the paragraph introduction) described in the Technical Design document.

**Environment**
Interesting details about the event-collection implementation are:
- JMX is an integral part of IBM Websphere; all admin functionality is realized through JMX. The location of the MBean-server determines the implementation of connectors with the event-monitoring iteration.
- Special attention goes to the registering of notifications and MBean’s on the MBean-server and the time of registering, this is crucial for a good implementation of event-generation and event-collection.
- The namespace of an MBean is controlled by the KeyNamingStrategy of an MBean.

#### 6.4.2 WMI implementation
The end result of the event-collection implementation for WMI. The full implementation is (like stated in the paragraph introduction) described in the Technical Design document.

**Environment**
Interesting details about the event-collection implementation are:
- WMI is an integral part of Windows 2003, every instance of Windows has a separate WMI registry.
- WMI .NET offers an implementation of a connector with the event-monitoring iteration.
- ASP.NET 2.0 Health monitoring is an integral part of ASP.NET 2.0.
7. Event-generation and Event-collection: Realization

A description of the realization of event-generation and event-collection, experiences with this realization and an evaluation with the target group is discussed here.

7.1 Experiences

The experiences of realizing this iteration are discussed via a number of Proof of Concepts. These Proof of Concepts give an overview of the lessons learned and techniques used to realize the concepts stated in the Technical Design. Next to the Proof of Concepts a load test regarding the overhead of event-generation onto the operation of Web Services is described.

7.1.1 List of Proof of Concepts

The following list of Proof of Concepts is related to the realization of event-generation and event-collection:

- The experimentation with Spring .NET AOP and Spring J2EE AOP, the making of example Web Services with AOP proxy factories and advices.
- The making of a .NET Web Service that sends WMI events to WMI via the WMI .NET implementation.
- The making of a .NET Web Service that sends WMI events to WMI via ASP.NET 2.0 Health Monitoring.
- The making of a JMX standard, dynamic, model en open MBean implementation into a J2EE Web Service to experiment with a JMX management interface.
- The making of a sample implementation of a JMX MBean that can send notifications to an MBean Server within Websphere Application Server 5.1 via the usage of IBM Websphere JMX Libraries and standard JMX libraries.

7.1.2 Load test

A load test has been done to investigate the overhead of event-generation on the normal operation of a chain of Web Services. One of the demands stated in the Technical Design (Chapter 6.1.2) mentions that the overhead of AOP and event-generation cannot influence the performance of a chain of Web Services drastically.

To meet this demand the performance of a chain of Web Services with event-generation and AOP enabled and the performance of that chain without event-generation and AOP need to be measured and compared to each other.

This is done in a load test that is performed on a simplified chain of 2 (both a .NET and J2EE Web Service) Web Services, the complete analysis and results of this load test are described in a separately supplied document, a summary of this analysis and results will be given here.

Analysis

The analysis of this load test is done via 4 scenarios. These scenarios differ in number of concurrent users and think time per user to give a more complete view of the performance of a Web Service under a certain load. The 4 scenarios are played for both chains of Web Services (with and without event-generation); the run-time environments of those chains are identical.

Results

Three out of 4 scenarios showed an overhead of AOP of 3 to 7 %, the heaviest scenario showed an overhead of about 30%. The amount of errors (time outs of the run-time environment) is also measured during the recording of the scenarios and is not noticeable higher for the Web Services with event-generation.
Conclusion
The conclusion of this load test is:
- More, and more specific load tests have to be done to exactly measure the overhead of event-
generation and AOP on a Web Service.
- The chain of Web Services used for this load test consists out of dummy Web Services with a
minimal execution time, overhead of event-generation will thus be maximal because of the small
normal execution time of the Web Service, and overhead of event-generation on normal Web
Services will likely be a number of times smaller.
- The performance of the whole chain of Web Services is measured, the individual J2EE and
.NET performance is not measured, run-time environment information is not taken into account,
and thus nothing can be said about possible bottle necks of .NET or J2EE or run-time
environments.

7.2 Evaluation
An evaluation of the results of this iteration. For a description of the total evaluation see Chapter 2.6.3.

7.2.1 Approach
A review of the results of these iterations is a verification of the implementation and realization (Functional,
Technical Design and Realization) against the chosen approach or solution stated in Chapter 2.5 and 3.
The approach for this verification is as follows:
- Find the suitable persons for the evaluation of this iteration.
- Discuss the results of these iterations with them, through discussing the following list of items:
  o Description of the iterations event-generation and event-collection.
  o Demands and wishes for the implementation of event-generation and event-collection.
  o The elaboration of the concepts into the Functional Design.
  o The elaboration of the concepts into the Technical Design.
- Through asking questions about the results of the implementation and realization and the
relationship between these results and the chosen approach or solution the evaluation of these
iterations is done. Besides these questions a large part of the evaluation consists of a little bit of
self reflection.

7.2.2 Evaluation results
The results of the evaluation done for the first and second iteration event-generation and event-collection.

Evaluation persons
The evaluation of these iterations is done with the following persons:
- Duncan Doyle
  Duncan Doyle is a J2EE architect within the CC J2NET; he is well qualified for the technical
  side of J2EE. Besides his part in the evaluation of this graduation project Duncan is associated
  with this graduation project as a technical assistant for J2EE.
- Saber Karmous
  Saber Karmous is a .NET architect within the CC J2NET; he has a lot of experience with .NET
  and is associated as an architect with a lot of projects that have delivered .NET Web Services
  within ICT. Besides his part in the evaluation of this graduation project Saber is associated with
  this graduation project as a technical assistant for .NET.

Evaluation material
The material that is discussed during this evaluation can be found in the chapters 3, 4, 5 and 6.
Verification
The end results of the verification of the 1st and 2nd iteration.

Approach
The iterative approach, the separation in the way that every iteration is handled apart from the other iterations and the requirements engineering have given a clear view on the demands and wishes of these iterations.
Prince2 is responsible for the management of the iterations, by making clear at the start of the iteration what the deliverables for each iteration were, these deliverables could be managed better.
A separation between the Functional and Technical Design is difficult, functionality overlaps with technical details, a little bit of intuition and subjectivity is required to distinguish functional and technical details. A good end result or a deliverable is not easily guaranteed.
Working in iterations makes thinking about the following iterations necessary. If this next iteration is not yet fully prepared it is not easy to say something useful with regards to that iteration. A next iteration can not be started when a previous iteration is not yet finished but this next iteration needs to be taken into account and that is difficult.

Implementation and realization
The different approaches while implementing the Functional and Technical Design, mentioning these approaches explicitly in the concerned documents have helped to make demands and wishes clear. By focusing on those approaches a funnel effect has been created, the deliverables have been reached and realized in a progressive way.
All choices made in the Functional and Technical Design are well argued because of the way requirements, demands and wishes are handled.
The realization of concepts stated in the Technical Design through a number of Proof of Concepts have helped me to get familiar with the (for me) relatively new environment of .NET and J2EE technologies. When pieces of functionality or technique are separated the realization way of these techniques and functionality can be viewed in a purposeful way. A Proof of Concept takes care of the experience needed to find a generic, feasible solution to realize a technique or a piece of functionality.
The complete realization and testing of the deliverables of the 1st and 2nd iteration is done through the application of these deliverables on a project that realized both .NET and J2EE Web Services. These Web Services were part of several chains of Web Services that existed in both FrontOffice and BackOffice.
By applying the deliverables of these iterations directly onto a suitable project the feasibility, practical implementation, points of attention during the implementation of event-generation for both new and existing Web Services and many other things could be tested and noticed, which helped to raise the quality of the deliverables of event-generation and event-collection.

Results / deliverables
The generation of events via the implementation of Aspect Oriented Programming is well founded. By thinking about and putting focus onto functionality and technique a well considered and implemented realization is made.
The usage of WMI and JMX for the generation and collection of events is a good choice; both concepts are a standard for the generation and distribution of application management and health information.
The signal implementation meets the demands and wishes for the definition of performance, availability and error information.
Part 3: Event-monitoring
8. Event-monitoring: Preparation

This chapter describes the preparation done for the event-monitoring iteration. The chosen approach, scope and target group is explained. This chapter forms the basis of this iteration.

8.1 Approach for the 3rd iteration

The 3rd iterations will be filled in as follows:

- Preparation (see this Chapter):
  - Definition of scope and target group of event-monitoring.
    - The scope and target groups of the global approach are translated to scope and target groups of the current iteration.
  - Description of the demands for the iteration of event-monitoring.
    - The demands, starting points and critical factors that are known and exist with regards to a possible realization of event-monitoring are stated here.
  - Projects concerning chain monitoring and event-monitoring within Fortis ASR
    - A summary of all projects within ICT that can concern chain monitoring or event-monitoring.

- Literature study (see Chapter 9):
  - Preparation of the literature study.
    - A short preparation for the literature study is described here.
  - Literature study and results of this literature study
    - Literature study to event-monitoring plus the results of this study.
  - Conclusion
    - Conclusion of this literature study.

- Function Design (see Chapter 10):
  - In this phase the following steps will take place:
    - The demands that are formulated within the literature study are translated to demands for the functional design, conversations with ICT an FAB will take place about those demands.
    - The concepts that will form the basis of event-monitoring derived from the literature study are examined and are further elaborated in the Functional Design; this elaboration uses the demands formulated at the beginning of the Functional Design.
    - After elaboration of the concepts on functional level, gearing these concepts with ICT architects and FAB will take place. With these architects and FAB validation of these concepts will take place to see if the demands that were formulated at the beginning of the Functional Design are met.
      - The way this validation will take place will be described in the Functional Design; the elaboration of this validation will be described in a separate validation document.

- Technical Design (see Chapter 11):
  - The demands that are formulated within the literature study are translated to demands for the technical design, conversations with ICT an FAB will take place about those demands.
  - The concepts that will form the basis of event-monitoring derived from the Functional Design are examined and are further elaborated in the Technical Design; this elaboration uses the demands formulated at the beginning of the Technical Design.
  - After elaboration of the concepts on Technical level, gearing these concepts with ICT architects and FAB will take place. With these architects and FAB validation of these concepts will take place to see if the demands that were formulated at the beginning of the Technical Design are met.
    - The way this validation will take place will be described in the Technical Design; the elaboration of this validation will be described in a separate validation document.

- Realization of the Technical Design and delivery of documentation concerning the realized products will take place.
The above interpretation is abstracted out of the following items:
- Necessary interpretation of a graduate project within the graduation direction of Software Engineering in the direction of Computer Science at the University of Technology in Delft.
- The choice of standard management method within ICT. This standard management method is Prince2 and contains the formulation of a Functional Design and Technical Design and the delivery of documentation concerning the implemented products.

8.2 Scope
The scope of event-monitoring is translated from the global scope stated in Chapter 2, together with the global approach defined in Chapter 2.5 an application onto the iteration of event-monitoring is made here.

The scope of event-monitoring is described as the monitoring of chains of .NET and J2EE Web Services for their health. Health can be subdivided into different areas:
- Performance
- Availability
- Errors

The application area of third iteration consists of information collected by the .NET and J2EE Web Services that are suited with event-generation. This information needs to be translated to visual information about the actual health of the Web Services and chain of Web Services.

8.3 Target group
The target groups and their interests in event-monitoring are described here. A translation from target groups and global approach onto the iterations of event-monitoring is made here.

The target groups and their interests in event-monitoring are:

ICT
A subdivision into two departments can be made here:
- Exploitation & Infrastructure (now known as Information Technology):
  This department is mainly concerned with Operation Control, contents of web services are not important in this area. Main concerns are:
  o Does a web server run well?
  o Is a web service available?
  o Global messages that indicate the health of a web service in general.
- System Development (now known as Information Services):
  This department is mainly concerned with the development of the web services and therefore wants to know more about the health of the web services:
  o How well does a web services perform?
  o Is a Web Service available all the time?
  o How many errors are generated by the web service?

FAB
Functional Application Management (shortly FAB) is a department that is part of Fortis ASR and exists of functional administrators. The main concern of this department is the people that use the software made by ICT.
This party is concerned with chain monitoring in a way the users of the software are concerned with the software. This party is interested in chain monitoring on a message level, to trace a message through a chain of web services, to see what has happened with the message.
They are answerable towards intermediaries and customers (of those intermediaries) through questions and reports.
8.4 Demands and wishes

The starting points and critical factors that are linked to the event-monitoring iteration. These starting points and critical factors are the basis of the literature study to event-monitoring and the literature results as well as the conclusion of this literature study.

Demands and starting points / critical factors

The demands for this iteration are the demands of the target groups that are stated at the start of this graduation project translated towards demands for the target groups of this iteration. These demands form the starting points of this iteration. These demands are critical factors for this iteration because the success of this iteration is depending on the fulfillment of those factors.

8.4.1 Achieving starting points / critical factors

The starting points and critical factors are divided into common and run-time demands that are translated from the global demands stated in Chapter 2.4. Besides the global demands more specific monitoring demands that are pointed towards the monitoring and presentation of health information about the operation of Web Services and chains of Web Services exist.

The critical factors and starting points of event-generation are determined through:

Common / run-time factors:
- Conversations within ICT, conversations with architects, supervisors and developers. [6, 23, 24]
- Translation of the reason and scope (see Chapter 8.2) in concrete demands / wishes.

Monitoring:
- Conversations with FAB, the intention of these conversations was: what does FAB expect of the signals, to whom do they carry which responsibility. [23, 24]
- Conversations with ICT, the intention of these conversations was: what is of importance to monitoring of Web Services.

8.4.2 Starting points / critical factors

The demands for this iteration are separated by their origin from a target group. The list of critical factors / starting points for event-monitoring of Web Services contains the following points:

Common:
- Existing architecture of Fortis ASR hardware and software can not be essentially adapted.
- Costumer information can not be dealt with in the events.

FAB:
- The business process that is implemented through a chain of web services must be sub divisible in steps. These steps can be mapped on functions that are executed by web services.
- Steps of a business process can be executed in a parallel and sequential manner.
- The monitoring of a business process can be seen as a traffic light, a business process is healthy (well performance, full availability, no errors -> green traffic light), not so healthy or sick depending on the (amount of / percentage of) performance, availability and errors.
- The monitoring of a business process is a real-time process. Monitoring must take place in a continuous way. Performance, availability and errors information of the web services that implement a business process must be monitored real-time.
- Monitoring of a business process on message level is desirable. The tracing of a message through a chain of web services means that this department is answerable to questions from intermediaries and consumer about their individual transactions.
Exploitation & Infrastructures / Information Technology:
- Information / events that give information about the hardware or Operating System of a web server are of importance to this department.
- A periodically generated signal per Web Service about the health (a composition of performance, availability and errors) of a Web Service is desirable.
- The amount of traffic between the DMZ and the LAN-area has to be limited as much as possible.
- The traffic between the DMZ and the LAN-area has to meet security guidelines.

System Development / Information Services:
- All realized software in this iteration has to be made according to the applied standards within Fortis ASR. The standards are part of the software development process.
- Per chain the storage of events has to take place as centrally as possible, to create a central source of information for the event-monitoring tools.
- Implementation for both J2EE and .NET Web Services cannot be drastically different.
- Front office applications need to be realized with Microsoft .NET technology.

8.4.3 Heritage of previous iterations
Important factors from previous iterations:
- The iterations event-generation and event-collection have resulted in a possible list of connectors for the storage of events; these connectors need to be dealt with during the design and implementation of this iteration.
- Per Web Service events about performance, availability and errors are generated, these events are the starting point of the performance, availability and errors of the whole business process / chain.

8.5 Projects concerning ‘event-monitoring’ within ICT
Within the organization of Fortis VNL and ICT chain monitoring or a closely related form of monitoring has already been realized in several projects, this paragraph is a summary of those projects and the outcomes of those projects, the influence of every project on this iteration is also discussed.

8.5.1 Proof of Concept: BPM
Recently a proof of concept with IBM software concerning Business Process Management has started within the CC J2NET. The main target of this software is connecting business processes with web services, the execution and description of web services. This PoC is started while this literature study was written and has not yet been finished, therefore the results of this PoC will not be taken into account in this literature study. Some facts about this PoC are:
The implementation of BPM within the IBM Software consists of:
- A business process server in which business processes are specified and stored using BPEL4WS and BPEL4People.
- An application server with web services running on it.
- A run-time server on which the processes are coupled onto the web services.

8.5.2 Project ‘Chain monitoring’
An initiative started by ICT to offer the customer chain monitoring for all their information systems and chains. This project is just starting and is run by Gert Jan van Halem. Because of the starting phase in which this project resides its influences and consequences will not be taken into account in this iteration.
9. Event-monitoring: Literature study

The literature study to event-monitoring, the preparation, the results and the conclusion of that literature study. The literature study to event-monitoring is the basis for the functional and technical design of event-monitoring.

9.1 Preparation

The assignment in more detail; this description is based upon the starting points / critical factors described in the previous chapters. The target of this chapter is to formulate the starting points for the concepts discussed in the following chapters as clearly as possible.

9.1.1 Basis

The basis of this iteration is made in the previous 2 iterations event-generation and event-collection. The starting point of this iteration is the fact that information about performance, availability and errors of a chain of web services must be built upon the information about performance, availability and errors of the individual web services. This information from the individual web services does not contain any chain-specific information and does not hold information about connections between web services. The basis of this iteration is generated performance, availability and error information from the individual web services. This generated information is made available for storage through the event-collection iteration.

9.1.2 Definitions

Subject of this literature study are web services and especially for this iteration business processes, the definitions of both terms are stated beneath:

**Business process:**

Wikipedia: A business process is a collection of related structural activities that produce something of value to the organization, its stakeholders or its customers. It is, for example, the process through which an organization realizes its services to its customers. [26]

Webopedia: A specific event in a chain of structured business activities. The event typically changes the state of data and/or a product and generates some type of output. Examples of business processes include receiving orders, invoicing, shipping products, updating employee information, or setting a marketing budget. Business processes occur at all levels of an organization’s activities and include events that the customer sees and events that are invisible to the customer. [25]

**Web Services:**

IBM Infocenter: Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains. They can be local, distributed, or Web-based. Web services are built on top of open standards such as TCP/IP, HTTP, Java, HTML, and XML. ... [1]

Lucent: Open standard (XML, SOAP, etc.) based Web applications that interact with other web applications for the purpose of exchanging data. Initially used for the exchange of data on large private enterprise networks, web services are evolving to include transactions over the public Internet. [2]
9.1.3 Starting point of the literature study

Demands of the business / FAB
Business processes that consist of web services need to be monitored to make clear that the business processes is executed with a certain performance and availability and no errors.

Demands of ICT
Web Services are the deliverables of ICT. The distinction of web services within business process is important because ICT develops web services and not business processes. ICT can therefore only take care of their deliverables and wants to monitor web services instead of a complete business process. Interesting views are the performance, availability and errors of individual web services.

Interesting fact
The first conversation with FAB made clear that a business process can be divided into steps. Web Services can be mapped onto those steps.

Objective of the literature study
Business and FAB demand the monitoring of a business process, ICT demands the monitoring of individual web services. The health of the individual web services was the essence of the iterations event-generation and event-collection. To provide information about the health of a business process a way needs to be found to combine health information of individual web services to obtain health information of a business process.

9.2 Literature study
The chosen approach for the literature study, how the literature results are obtained and the results are described in such a way that a conclusion about the best suitable concept for this graduation project can be drawn in the next chapter.

9.2.1 Approach
As a result of the previous paragraphs and chapters a literature study has been done to event-monitoring. The found literature has to fulfill the demands, starting points and critical factors (see Chapter 8.4) and an implementation of the mechanisms mentioned in the literature has to fit within the existing architecture and infrastructure (see Chapter 1.4).

The found literature will be recorded in this report via:
- A short but clear description of the literature and possible implementation in the existing architecture and infrastructure.
- An application of the literature onto the existing architecture and infrastructure and an overview of the possibilities and impossibilities considering the demands, starting points and critical factors stated in Chapter 8.4.
- An evaluation of the literature, consideration of the reliability of the technology, best practices and possibilities of the mentioned solution in the realization phase.

The results of the literature study are shown in the following paragraphs:
9.2.2 Web Services Composition
9.2.3 Workflow Management
9.2.4 Business Process Modeling and Management
9.2.2 Web Services Composition

The idea behind Web Services Composition

Web Services are pieces of software that can execute elementary functions / functionality. These pieces of software are made in a way that these pieces can be called from over a network. [29]

A Web Service contains only limited functionality, a web service can execute only a single function whether a business process normally exists of a combination of functions.

Web Services Composition is a concept that takes care of the combining of multiply functionalities of multiple Web Services. This combining is done through a process model named Web Services Composition Language. This process model ensures that the functionality offered by the various Web Services is combined in one composite Web Service. [27]

Through a technique like WSDL and several other techniques it is possible to connect individual Web Service in a uniform way to a process model. The figure below shows the position of a Web Services Composition Language relative to various other key technologies in the specialized area of Web Services. The focus of Web Services Composition is on combining functionality offered by different individual Web Services. This way functionality is bundled and can be offered like one Web Service. It is easier to call one Web Service instead of several Web Services. The relations between the combined Web Services and the message flow between those Web Services stays hidden from the calling party because they can only see one Web Service.

![Web Service Composition Diagram](image)

Event-monitoring – Application of Web Services Composition

The concept of Web Service Composition distinguishes individual Web Services from combined functionality offered by those individual Web Services through the usage of a Web Services Composition Language. The combined functionality contains a process model; this process model specifies the order in which the individual Web Services are executed. This process model can be seen as some sort of business process. A business process also contains a number of elementary steps that are combined to a business process through some sort of process model. The concept of Web Services Composition thus offers the possibility to specify and model a business process and clearly distinguish Web Services from the business process they partly implement.

Latest conditions– Web Services Composition

A lot of different Web Services Composition Languages exist; a real standard is not yet defined because of the large numbers of markup languages that exist in this area. The overview below shows a number of markup languages, their weak and strong points and the area of application of the specified language are mentioned:

- XLANG from Microsoft

  XLANG is developed by Microsoft as a language to model business transactions. Business transactions are large and long lasting interactions between Web Services. The emphasis with XLANG is on the visibility of the message traffic between Web Services. The emphasis has the disadvantage that XLANG uses a somewhat unusual interpretation of WSDL. [28]
Web Services Flow Language from IBM
WSFL is a language developed by IBM. The strengths of this language are the specification and implementation of a business process. The main weakness is the ability to model a business process. [28]

Business Process Execution Language 4 Web Services (BPEL4WS)
BPEL4WS is a combination of XLANG from Microsoft and WSFL of IBM. Through the combination of specific properties of both languages a language has been created that can model executable and abstract processes. [28] BPEL4WS makes use of a number of specifications developed by IBM and Microsoft to complete the Web Service technology.

Darpa Agent Markup Language and Ontology Web Language
DAML and OWL are initiatives of a number of parties (DAML committee, W3C and WebOnt) to specify the ontology or domain of Web Services in a basic language coupled with a number of extensions. By enriching a number of old standards in the domain of Web Services, a language is created that tries to cover the whole domain (description, discovery and combination) of Web Services. [28]

Application within this graduation project – Possibilities and impossibilities
In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment and reason of this graduation project.

Possibilities:
- The modeling of a business process is part of Web Services Composition
  Modeling a business process, a process that is realized through amongst others Web Services is possible with a Web Services Composition Language.
- Distinguish individual Web Services
  Web Services Composition distinguishes individual Web Services from a business process. By distinguishing individual Web Services from combined functionality provided through those Web Services it is possible to model a business process that consists of individual Web Services.

Impossibilities:
- The goal of Web Services Composition is to combine elementary functionality of different Web Services into a combined Web Services that offers all that functionality. The goal is not to model a business process that consists of different Web Services.
- A real Web Services Composition Language standard does not yet exist.
  A lot of different Web Services Composition Languages exist; a real standard language that offers all the functionality wanted from a Web Services Composition Language has not yet been defined. This does make the choice which language and functionality to choose a little bit harder.

Evaluation
Web Services Composition is more or less the melting of Web Services to realize certain functionality. SOA tries to realize just that, functionality is realized through a chain of Web Services. Web Services Composition looks to be the right concept in that way.
The modeling of a business process however is only a part of the concept Web Services Composition; the modeling of a business process is one of the more common offered functionalities in a Web Services Composition Language.
A lot of Web Services Composition Languages are already available for production use, a real standard is yet still missing, the differences are however minimal.
9.2.3 Workflow Management for Web Services

The idea behind Workflow Management for Web Services

A workflow describes the order of a set of tasks that are carried out by different agents to complete a given procedure within an organization. Depending on the computer technology workflow has become automation or semi automation within the process of completing procedures. According to a set of process rules a number of documents, information and tasks can be divided over different agents. [31] Workflow management consists of the management of such a workflow or business process. Other important parts of workflow management are the definition and validation of such a process and the managing and monitoring of the execution of a workflow process.

Workflow management is a relatively old concept that has been applied many times; the application onto Web Services however is relatively new and not that old. Workflow management for Web Services is a specialization of workflow management in the direction of E-Services and E-Business; Web Services represent the many tasks of the business processes behind these terms. A few initiatives have been taken to develop a workflow management system that can manage workflow processes that consist of Web Services [30, 31, 32 and 33] (Figure 14 shows a reference model of a workflow management system).

![Reference model of the Workflow Management Coalition (©WfMC). [34]](image)

Event-monitoring – Application of Workflow Management for Web Services

Workflow Management Systems to manage Web Services contain the possibility to connect a business process to Web Services. A workflow management system contains the representation of a business process. A workflow management system however is fully equipped to manage, assist and monitor the execution of a business process. The scope of a workflow management system is much larger than the scope of the assignment formulated in Chapter 9.1.
Chain monitoring for Web Services

Latest conditions – Workflow Management for Web Services
Application of Workflow Management onto Web Services takes place through a number of initiatives, an
enumeration beneath shows a number of those initiatives:
- WorkFlow Management Coalition [35]
  The WFMC is an organization that tries to promote and standardize workflow technology. Web
  Services and Business-2-Business integration is recognized as a new and important technology
  in the world of Information Systems. Attempts are made to define a language to model business
  processes in which web services attend.
- Worksco [30]
  Worksco is a framework with which workflow applications can be easily build.
  Worksco contains functionality to define a business process and to monitor the execution of this
  process.
- HP Process Manager HPPM [32]
  HPPM is a Workflow Management System that can deal with Business-2-Business Interactions.
  Business-2-Interactions ask for a special kind of aiming that is comparable to Web Services,
  tradition workflow however does not need this special kind of aiming.
- SemWebQ [33]
  SemWebQ is a framework that uses OWL as binding between Web Services and a workflow
  process. SemWebQ also has dynamic rule based engine to measure and judge the quality of the
  execution of a business or workflow process.

Application within this graduation project – Possibilities and impossibilities
In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment
and reason of this graduation project.
Possibilities:
- A lot of similarities between a business process and a workflow process
  A workflow process as used through a Workflow Management System has a lot of similarities
  with a business process; the ability to correctly model a business process is available inside
  workflow management as a concept.
- Initiatives with web services are yet available
  The initiatives to use Workflow Management Systems to manage Web Services are already
  started; the application of Workflow Management to Web Services is easier when those
  initiatives are already available.

Impossibilities:
- A real standard for using Workflow Management with Web Services is not yet available
  A real unambiguous standard to combine Web Services and Workflow Management is still
  missing, a number of initiatives on this area are in place but all use different implementations of
  Workflow Management.
- The focus of Workflow Management is on managing a business process
  Workflow Management Systems are focused on managing a business process, modeling a
  business process is only a small part of this process. The focus thus lies on managing and not on
  modeling, the focus of this iteration is mainly modeling.

Evaluation
Workflow Management is purely based on managing and monitoring the execution of a business process,
the modeling of a business process is just a very small part in this process. Workflow Management Systems
are already used to manage and monitor business processes that consist of Web Services. The application of
Workflow Management Systems onto Web Services is therefore not entirely new. The execution of a
business process is mainly done by the Workflow Management System and the emphasis lies on the
management of the business process and that is not conform the assignment formulated in Chapter 9.1.
9.2.4 Business Process Modeling and Management

The idea behind Business Process Modeling and Management

Business Process Modeling (BPMo) and Business Process Management (BPMa) are terms that are related to each other; they are used together and have a lot in common. Business Process Modeling is a set of standards and technologies to design and model a business process. Business Process Management is a way of supporting a business process through the usage of methods, techniques and software to design, execute, control, manage and monitor operational (business) processes within an organization. BPMo is a part of BPMa according to above descriptions of BPMo and BPMa, designing and modeling is a subset of managing.

Within organizations Web Services are more and more used to automate / realize (parts of) activities of a business process. These Web Services and accompanying business process must be managed and modeled; BPMa is responsible for these activities. Connecting Web Services and business process is however not the primary goal of BPMa, the primary goal is much more business-related and is the improvement of business processes within an organization.

Event-monitoring – Application of Business Process Modeling and Management

Through BPMo business processes can be modeled, Web Services are more commonly used to realize (parts of) activities in business processes so the possibility to connect Web Services with activities of business processes is available. BPMa is however much wider oriented and also has the ability to manage a business process. The goal of BPMa is mainly the improvement of business processes and that is a lot bigger than the assignment and goal formulated in Chapter 9.1.
Latest conditions – Business Process Modeling and Management
Application of BPMa en BPMo onto Web Services takes place on a large scale, a number of applications is enumerated below:

Business Process Modeling:
- Business Process Modeling Notation (www.bpmn.org) (BPMN) [39]
  An initiative of the Business Process Management Initiative (BPMI) to come to an unambiguous modeling notation of business processes.
- Business Process Execution Language (BPEL) [38]
  An initiative of OASIS to come to an unambiguous execution language for the execution of business processes.
- Business Process Modeling Language (BPML) [39]
  An initiative of the Business Process Management Initiative (BPMI) to come to an unambiguous modeling language of business processes.

Next to above enumerated applications of BPMo there are a lot more applications. The problem is however that here are too much initiatives towards BPMo languages, the initiatives do not differ very much but cannot be tolerated next to each other because of the lacking of a real standard. [38]

Application within this graduation project – Possibilities and impossibilities
In this section the possibilities and impossibilities of this concept will be discussed regarding the assignment and reason of this graduation project.

Possibilities:
- BPMo is suited for the modeling of a business process that uses Web Services to realize activities of that business process. BPMo is very well suited for the modeling of business processes and because of the fact that Web Services are seen as part of those processes the possibility to model Web Services within such a process is available.

Impossibilities:
- BPMa and BPMo are connected to each other
  BPMo and BPMa are heavily connected to each other, BPMo is a part of BPMa and BPMo is focused on supporting BPMa. BPMa and BPMo can therefore not be seen apart from each other. BPMo however suites the assignment described in Chapter 9.1 best, which in this context stated is a disadvantage.
- The goal of BPMa is to improve business processes
  The goal of BPMa is not focused on ICT, the goal of BPMa is focused on the improvement of business processes within an organization and this goal is not part of the assignment described in Chapter 9.1. This assignment is about the modeling of a business process and not to improve a business process.

Evaluation
BPMo and BPMa are widely spread areas, the focus mainly lies at the business, the improvement of business processes and less with ICT. This focus takes care of the fact that this concept is not well suited for just the modeling of a business process. Next to this fact there is a lot of movement on this area, also within Fortis Insurances Netherlands a number of initiatives are brought together to see what the application of BPMa and BPMo can do for an organization like this. BPMa is an interesting area of research but of less interest to this assignment because of the difference in focus and goal specified in Chapter 9.1.
9.3 Conclusion

In the preceding paragraph a number of concepts regarding event-monitoring have been described. Every concept is described via a short introduction and an overview of the possibilities and impossibilities as well as an application in the context of this graduation project. This paragraph forms a conclusion to this literature study, a decision is made which concepts are going to be used and implemented in this graduation project.

Choice – event-monitoring

Looking at the concepts for event-monitoring described in Chapter 9.2, putting together the possibilities and impossibilities, the following conclusion can be drawn.

Common

The concepts Web Services Composition, Workflow Management and Business Process Management and Modeling all have their connections with modeling a business process and the interface between Web Services and business processes. The goals and areas of focus of the different concepts are however far apart from each other. Web Services Composition focuses on the Web Services. Workflow Management tries to offer an all-in-one solution to manage a business process from a workflow point of view. The main goal of Business Process Management and Modeling is to improve business processes and the managing of business processes.

When we compare above goals and focus areas to the assignment specified in Chapter 9.1 then it is clear that the goal of Web Services Composition does best correspond to the goal of the assignment of Chapter 9.1. The focus with Web Services Composition is on the Web Services themselves, through a process model Web Services are combined and are offered like one Web Service.

Possibilities versus Impossibilities

On the basis of the list of possibilities and impossibilities per concept the tendency that is described in the above chapter can also be seen in the comparison of the list of possibilities and impossibilities. Web Services Composition is a good match to the assignment, does offer a challenge on choosing a Web Services Composition Language for modeling a business process / process model. Workflow Management has a larger goal and focus when compared to the assignment by focusing on the management of a business process, specialization in the direction of Web Services is however available. Business Process Management and Modeling focuses on the improvement of business processes and business processes themselves. BPMa is fully devoted to the improvement of business processes and that is clearly not what the assignment stated in Chapter 9.1 says.

Conclusion

On the basis of the possibilities and impossibilities and the concepts described in the ‘Common’ section of this paragraph the concept Web Services Composition offers the best opportunities to implement and realize the assignment described in Chapter 9.1.

Prospect

A Web Service Composition Language will have to be chosen that offers the possibility to both model a business process and offer the possibility to distinguish individual Web Services (that implement this process) from this business process. Above described part will be contained in a dashboard tool that will be realized for both FAB and ICT. This dashboard tool will give insight in the health (performance, availability and errors) of a chain of web services.

The Functional Design for event-monitoring, the approach and the design results of this Functional Design. The Functional Design to event-monitoring is the basis for the Technical Design of event-monitoring. The information described in this chapter is a summary, a detailed description of the information described in this chapter can be found in the separately supplied Functional Design document.

10.1 Approach of the Functional Design

Description of the approach taken by making the Functional Design, the approach is completed with the demands and wishes that form the basis of the Functional Design.

10.1.1 Approach

The approach that was followed during the making of the Functional Design is as follows:

- Focus on concepts.
  The focus during the making of the Functional Design was on the concepts found in the literature study and the implementation of those concepts that realized the functionality described in the demands and wishes.
- Functionality offered has focus
  The concepts chosen needed to be described in a way that the functionality offered was clear.
- Focus on generic and feasible solutions
  The implementation of concepts also needed to take care of the fact that a generic solution could be made. That solution needed to be feasible and realistic in terms of demands and wishes.

10.1.2 Demands and wishes

A description of the demands and wishes of the functional implementation of event-monitoring through the achievements of those demands and wishes and the actual demands and wishes is made here. During the achievement of the demands and wishes it became clear that a clear separation within event-monitoring between the storage part of the monitoring iteration and the presentation part of the monitoring iteration could be made. The demands and wishes are thus ordered by storage and presentation layer.

Achievement demands and wishes

The demands and wishes for the Functional Design of event-monitoring are determined through:

- Conversations with architects within ICT and with FAB.
- Translation of the demands and wishes of the graduation project and the literature study to event-monitoring to demands and wishes on a functional level.

Demands and wishes

The list of demands and wishes for event-monitoring contains the following points:

Common:

- Existing architecture and infrastructure of ICT hardware and software can not be essentially adapted.

Storage:

- Standards must be used as much as possible on the different areas of implementation. The application of chain monitoring on different development and production environments touches different parties and users and thus standards must be used to ensure the best solution is developed.
- Implementation for both J2EE and .NET Web Services cannot be drastically different.
- Storage of events per chain must take place as central as possible; this way a central source of information can be created. This source can then deliver all monitoring information needed.
- Information from the past must be stored too, the past can be seen as not real-time, so today is real-time and yesterday and last week etc is the past.
- Storage in combination with Microsoft .NET technology must be realized in combination with MS SQL Server 2000, this is a common development standard in CC J2NET.
Chain monitoring for Web Services

Presentation:
- A possibility to view monitoring data of a business process real-time must be created. Preferably in the realization form of a traffic light.
- A possibility to view monitoring data of Web Services real-time must be created.
- Frontoffice applications must be realized with the help of Microsoft .NET technology and must preferably be web-enabled.
- Real-time monitoring must be realized, monitoring data that is viewed must be real-time updatable.
- The possibility to correctly model a business process and the connection with Web Services in a modeling language must be created.
- The possibility of monitoring of a business process on a traffic message level is wanted, the tracing of traffic messages through a chain of Web Services means being able to justify the status of transactions initialized by end consumers and intermediaries.
- A possibility to view monitoring data from the past of Web Services must be created.

10.2 Presentation
The description of the event-monitoring presentation layer in a functional way.
The description consists of the following elements:
- Choice of modeling language
- Choice of presentation technique
- Separation presentation layer in FAB and ICT part
- Data model for the presentation layer

10.2.1 Choice of modeling language
The choice of a modeling language to model both a business process and the connection between a business process and Web Services. The choice is made through the definition of important factors, a comparison of the possibilities and the motivation of the final choice.

Factors
An overview of the factors that are of importance while choosing a modeling language and a motivation of those factors:

<table>
<thead>
<tr>
<th>Factor</th>
<th>Contents of factor</th>
<th>Motivation behind factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expressiveness</td>
<td>Expressiveness of a WSCL, semantics, formal model. A set of features that makes a WSCL easy to use, self documenting and elegant.</td>
<td>A WSCL must have a good expressiveness, good semantics because a business process should be clearly and in a good way expressed.</td>
</tr>
<tr>
<td>Tool support</td>
<td>The availability of enough tools to make the application of this WSCL possible in a good way.</td>
<td>Tools are important for the application of a WSCL, no tools available no good application of a WSCL.</td>
</tr>
<tr>
<td>Process modeling construct support</td>
<td>The possibility to apply different constructs in a WSCL to manage the flow of a business process.</td>
<td>A business process should be modeled in a correct, truth like way. To use as many constructs as possible by doing this is a must.</td>
</tr>
<tr>
<td>Acceptance within ICT</td>
<td>A short description of the consequences of using this WSCL within the CC J2NET. Possible (dis)advantages of particularly this language. A description of the consequences of demands described in the ‘Program of Demands’.</td>
<td>A standard WSCL is not yet defied within ICT or CC J2NET; initiatives on the area of BPM that use BPEL and BPML are already taken. Future initiatives on the area of Business Process Modeling will make use of these two known WSCL’s.</td>
</tr>
<tr>
<td>Best practices</td>
<td>Current applications and experiences of this WSCL.</td>
<td>Applications and experiences with this WSCL are important for the usability and future perspective of this WSCL.</td>
</tr>
</tbody>
</table>

Table 13: Description of factors that are of importance to the choice of a modeling language.
Comparison
Which possibilities for a modeling language exist: (per possibility a description of the important factors)

<table>
<thead>
<tr>
<th>Factor \ Tool</th>
<th>WSCI</th>
<th>BPEL4WS</th>
<th>DAML-S</th>
<th>BPML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expressiveness</td>
<td>Expressiveness is good, WSCL is easy to use.</td>
<td>By the merge of XLANG (MS) and WSFL (IBM) a very expressive languages has been created.</td>
<td>A special Web Service Ontology that is build upon the DAML + OIL ontology. DAML + OIL is an extension of RDF, XML and RDF-S. The expressiveness of DAML + OIL depends on the supported axioms.</td>
<td>To model a real business process in BPML is hard; certain areas of BPML are very well suited for this modeling; other areas are less well.</td>
</tr>
<tr>
<td>Tool support</td>
<td>Some editors are available on-line, but little is known of WSCI and SUN (the big player behind WSCI) has surprisingly little information available on their website.</td>
<td>IBM and Microsoft are concerned with BPEL and take care of the availability of enough tool support. No problems here.</td>
<td>A few commercial demos are available; a lot of other demos and portals are available but no production material.</td>
<td>No big enterprises that support this WSCL, also very little tool support.</td>
</tr>
<tr>
<td>Process modeling</td>
<td>More than enough constructs are supported.</td>
<td>More than enough constructs are supported.</td>
<td>More than enough constructs are supported.</td>
<td>A number of constructs are supported; the modeling of a complete business process however is not possible.</td>
</tr>
<tr>
<td>construct support</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acceptance</td>
<td>No definitive choice has been made about a business process modeling language within ICT, WSCI is neither a possible choice nor on the choice list.</td>
<td>A proof of concept is running on BPMa, BPEL is the main modeling language. BPEL4WS is an extension to BPEL pointed towards Web Services. Acceptance within ICT is already begun.</td>
<td>No definitive choice has been made about a business process modeling language within ICT, DAML-S is neither a possible choice nor on the choice list.</td>
<td>At ‘De Amersfoortse’ a BPM suite is installed that uses this language. ‘De Amersfoortse’ is however not ICT and separation of interests between these parties is a fact.</td>
</tr>
<tr>
<td>within ICT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best practices</td>
<td>Not much too find about it. This WSCL is more concerned with the choreography of Web Services than with business processes.</td>
<td>IBM applies BPEL in its BPM suite, a lot of other parties use BPEL as their WSCL / BPM language.</td>
<td>Not much too find about it. DAML-S is based on DAML + OIL and this ontology needs a much wider application than WSC.</td>
<td>Cordys BPM, used by ‘De Amersfoortse’ has no real results published yet, initiatives to continue are in place however.</td>
</tr>
</tbody>
</table>

Table 14: Comparison of possible modeling languages.
Final choice + motivation
The final choice of the WSCL to use depends on the weight of the factors. Which factors weigh the most and what about the demands issued in the ‘Program of Demands’.
‘Tool support’, ‘best practices’ and ‘Acceptance within ICT’ are important factors for the use of a WSCL within ICT. ‘Process modeling construct support’ and ‘Expressiveness’ are important factors for the application of a WSCL to model both a business process and the connection between a business process and Web Services.
When demands and factors are compared to possibilities for a WSCL, the following conclusion can be drawn:
- WSCI is not chosen because it has no real business process support, and there are no real usable best practices.
- BPEL4WS is supported by big parties like Microsoft and IBM, tool support is large. And the language offers the possibility to connect Web Services to a business process because this is a special variant of BPEL that is pointed towards Web Services.
- DAML-S and DAML + OIL are nice initiatives; the language DAML-S is very powerful.
- Composition of Web Services is however just a small part of the possibilities that DAML + OIL offers. The true power of the concept lies in the total connection between discovery, composition, execution and interoperability. The tool support and best practices are not that big.
- BPML is a language that has troubles to correctly and completely model a business process, this language is however already used within Fortis. Best practices are however not that big.
The final choice is BPEL4WS for the specialization towards Web Services and the wide Tool support.

10.2.2 Choice of presentation technique
The choice of a presentation technique to present the monitoring data towards the target groups. The demands and wishes are compared to a number of possibilities and a final choice is made.

Demands and wishes
Demands to the presentation technique are:
- A possibility to view monitoring data of a business process real-time, maybe in the form of a traffic light has to be created.
- A possibility to view monitoring data of Web Services real-time has to be created.
- Frontoffice applications must be realized with the help of Microsoft .NET technology and must preferably be web-enabled.
- Real-time monitoring must be realized, monitoring data that is viewed must be real-time updatable.
- The possibility to correctly model a business process and the connection with Web Services in a modeling language must be created.
- The possibility of monitoring of a business process on a traffic message level is wanted, the tracing of traffic messages through a chain of Web Services means being able to justify the status of transactions initialized by end consumers and intermediaries.
- A possibility to view monitoring data from the past of Web Services must be created.
Technology
Microsoft .NET v2.0 Technology in the form of ASP.NET 2.0 is the only possibility for the development platform because it is a standard within ICT for Frontoffice applications which this presentation dashboard tool is.

Real-time aspect of the presentation
The real-time aspect about the presentation of monitoring data expresses itself in a continuously updated interface in which the latest monitoring data is showed. This continuous updating of the interface means a lot of refreshing and interaction on the screen of the users. To ensure that this interaction and refreshing is directed and done in a proper way there are special framework available. These AJAX frameworks control the interaction that happens within a user interface.
There are a lot of AJAX frameworks but to keep it simple the framework that is already used within Fortis, the Microsoft ASP.NET Atlas Framework will be used in this graduation project. The functionality offered by this framework integrates well with ASP.NET 2.0 and is sufficient for the interaction supplied in the user interface.

10.2.3 Separation of presentation layer
The presentation layer is of importance to two target groups, FAB and ICT. These parties both have different interests and demands and wishes, therefore the presentation layer is split, to more easily meet the demands of both parties. The separation is as follows:
- FAB
  Focus lies on business process. Performance, availability and errors of the business process are important, monitor those health parts via a traffic light construction is what they want. The demand to trace a traffic message through a chain of Web Services also exists but is not realistic since the facilities on different levels of technique are not in place within ICT.
- ICT
  Focus lies on Web Services. Monitoring performance, availability and errors of individual Web Services in a real-time way is important

Next to this separation there exists the need for a historical view of monitoring data, this need is handled separately in the Technical Design because the demands and wishes for this kind of interface are very different from the above standing more important needs. The focus firstly lies on real-time monitoring and the monitoring of historical monitoring data is worked out separately on a technical level.

10.2.4 Data models of the presentation layer
A description of the data models that will be used within the presentation layer. The data models form the backbone of the Functional and Technical Design of the presentation layer. These data models are made according to the demands and wishes of this Functional Design.
The following 2 data models will be discussed:
- Global data model
  This data model discusses the important data and data layers for the presentation layer, the relations between the data are described. See Figure 16 for this data model.
- BPEL4WS data model
  BPEL4WS is used in the presentation layer to model a business process and the connection between that process and Web Services. The data model of this WSCL is important to the construction of the presentation layer. The connection between this data model and the global data model has to be made. The goal of this presentation layer is the mapping of the monitoring data to the visual presentation of the health of a business process represented by a chain of Web Services. See Figure 17 for the data model of BPEL4WS.
Figure 16: Global data model of the presentation layer

Figure 17: BPEL4WS data model
10.3 Storage
The description of the event-monitoring storage layer in a functional way. The description consists of the following elements:
- Choice of storage technique
- Data model for the storage layer

10.3.1 Choice of storage technique
A description of the choice of a storage technique to store the monitoring data. The demands and wishes are compared to a number of possibilities and a final choice is made.

Demands and wishes
Demands to the storage technique are separated by origin; the following list contains all those demands.

Common:
- Existing architecture and infrastructure of ICT hardware and software can not be essentially adapted.

Storage:
- Standards must be used as much as possible on the different areas of implementation. The application of chain monitoring on different development and production environments touches different parties and users and thus standards must be used to ensure the best solution is developed.
- Implementation for both J2EE and .NET Web Services cannot be drastically different.

Event-monitoring:
- Storage of events per chain must take place as central as possible; this way a central source of information can be created. This source can then deliver all monitoring information needed.

ICT:
- Storage in combination with Microsoft .NET technology must be realized in combination with MS SQL Server 2000, this is a common development standard within CC J2NET.
- Information from the past must be stored too, the past can be seen as not real-time, so today is real-time and yesterday and last week etc is the past.
- The traffic between the DMZ and the LAN-area has to meet security guidelines, the traffic must be validated by a Vordell fire wall, and the best way to validate this traffic is to make use of SOAP-messages over a HTTP-channel.

Description of the situation
Events that are generated by (the execution of) Web Services are gathered inside WMI and JMX. These events need to be transported towards a central storage point.

Possibilities
The storage of events thus consists of the processing and physically storing of all generated WMI and JMX events in a database. This processing and storing of WMI and JMX events can take place in 2 ways:
- Centrally, the processing and storage happens as close as possible to the database. (See Figure 18).
- De centrally, the processing and storage happens as close as possible to the web server on which the Web Services are hosted. (See Figure 19).
Figure 18: Processing centrally

DMZ / LAN | BUSINESS-domain | LAN
---|---|---
Web Server | WMI / JMX | Storage and / or Web Server
Application | Events | Processing and storage agent
| Events | Monitoring data | Database

Figure 19: Processing de centrally

DMZ / LAN | BUSINESS-domain | LAN
---|---|---
Web Server | WMI / JMX | Storage Server
Application | Events | Monitoring data
| Events | | Database
Beneath shown table shows a comparison between the ways of realizing the storage technique:

<table>
<thead>
<tr>
<th>Central processing and storage of events</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- The storage of events can be managed centrally.</td>
<td>- Not many possibilities towards validating the message traffic.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Not many possibilities towards regulating and controlling the amount of traffic between DMZ and LAN.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>De central processing and storage of events</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- A lot of possibilities towards validating the message traffic.</td>
<td>- The storage of events cannot be managed centrally.</td>
</tr>
<tr>
<td></td>
<td>- A lot of possibilities towards regulating and controlling the amount of traffic between DMZ and LAN.</td>
<td></td>
</tr>
</tbody>
</table>

Table 15: Advantages and disadvantages of concepts for the implementation of the storage layer of event-monitoring.

Choice
The eventual choice is the de central processing and storage of events. Because this way the message traffic between web servers and the storage database can be controlled and regulated. And this way the best solution regarding the security and demands that concern the infrastructure can be chosen.

10.3.2 Data model of the storage layer
The data model that will be used within the storage layer. This data model (see Figure 20) forms the backbone of the Functional and Technical Design of the storage layer. This data model is made according to the demands and wishes of this Functional Design.

The signal definition as defined in the Technical Design of event-generation and event-collection is being transferred via a one-on-one transformation to the storage layer because all of this information is relevant for the real-time monitoring of a business process and the underlying Web Services. A one-on-one transformation is a way to not lose any information.

Figure 20: Data model of storage layer
10.3.3 Monitoring data history

Next to the choice for storage technique and the description of a data model for the storage layer the demand to view monitoring data of the past needs to be worked out here.

Monitoring data is information from the past, this information must be stored too, the past can be seen as not real-time, so today is real-time and yesterday and last week etc is the past.

The moment a day ends the data of yesterday is monitoring data of the past, so every day the monitoring data of yesterday needs to be stored so that historical monitoring data is available.

Through the demands and wishes of both FAB and ICT on this area the decisions about the storage format and the need for a transformation is made.

Demands and wishes

Demands and wishes of FAB and ICT with regards to monitoring data of the past are:

- Important information with regards to performance signals of the past is:
  - The information is summarized per Web Service.
  - The information is summarized per hour.
  - The number of completed and not completed requests per Web Service is of importance.
  - The average response time per Web Service is of importance.

- Important information with regards to availability signals of the past is:
  - The information is summarized per Web Service.
  - The information is summarized per hour.
  - The percentage availability per Web Service is of importance.

- Important information with regards to error signals of the past is:
  - The information is summarized per Web Service.
  - The information is summarized per hour.
  - The information contains the number of same error messages.
  - The information contains the text of the error message.

Realization

Per day a summarization job needs to run over the monitoring data of yesterday, this job needs to fulfill the demands and wishes of above list. The summarization job needs to store the summarized monitoring data in such a format that above listed information is stored.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Level</th>
<th>Parameters</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Service method</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeNaam</td>
<td>Name of the executed service method</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeDuur</td>
<td>Number of execution msec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeVoltooid</td>
<td>Boolean (completed successfully)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeDatum</td>
<td>Date + time of generation of event</td>
</tr>
<tr>
<td>Task /</td>
<td>Service method</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td>Transaction</td>
<td></td>
<td>ServiceMethodeNaam</td>
<td>Name of the executed service method</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TaakNaam</td>
<td>Name of executed task / transaction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TaakDuur</td>
<td>Number of execution msec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TaakVoltooid</td>
<td>Boolean (completed successfully)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TaakDatum</td>
<td>Date + time of generation of event</td>
</tr>
<tr>
<td>Availability</td>
<td>Web Service</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceDatum</td>
<td>Date + time of generation of event</td>
</tr>
<tr>
<td>Error</td>
<td>Service method</td>
<td>ServiceNaam</td>
<td>Name of the executed Web Service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeNaam</td>
<td>Name of the executed service method</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeFoutBoods</td>
<td>Error message</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chap</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ServiceMethodeDatum</td>
<td>Date + time of generation of event</td>
</tr>
</tbody>
</table>

Table 16: Technical signal implementation used for the summarizing of monitoring data.
11. **Event-monitoring: Technical Design**

The Technical Design to event-monitoring, the approach and the design results of this Technical Design. The Technical Design to event-monitoring is the basis for the realization phase of event-monitoring.

11.1 **Approach of the Technical Design**

The approach taken by making the Technical Design, the approach is completed with the demands and wishes that form the basis of the Technical Design.

11.1.1 **Approach**

The approach that was followed during the making of the Technical Design is as follows:

- **Focus on techniques.**
  
  The focus during the making of the Technical Design was on the techniques that realize the concepts described in the Functional Design and the best implementation of those techniques.

- **Focus on generic and feasible solutions.**
  
  The techniques also needed to take care of the fact that a generic solution could be made. That solution needed to be feasible and realistic in terms of demands and wishes.

11.1.2 **Demands and wishes**

A description of the demands and wishes of the technical implementation of event-generation and event-collection through the achievements of those demands and wishes and the actual demands and wishes is made here. The common demands and wishes are described here; the more technique specific demands and wishes are described in the paragraphs describing the implementation of those techniques.

**Achievement demands and wishes**

The demands and wishes for the Technical Design of event-monitoring are determined through:

- **Conversations with architects within ICT and with FAB.**

- **Translation of the demands and wishes of the Functional Design and the literature study to event-monitoring to demands and wishes on a technical level.**

**Demands and wishes**

The list of demands and wishes for event-monitoring contains the following points:

**Common:**

- Existing architecture and infrastructure of ICT hardware and software can not be essentially adapted.
11.2 Presentation layer

The implementation of the presentation layer or dashboard tool is discussed; this paragraph is a short summary of the whole technical implementation described in the Technical Design document separately supplied with this thesis paper.

11.2.1 Approach

The approach that was followed during the making of the implementation of the presentation layer was:

- Define the different views that must be realized for viewing the monitoring data.
- Translate the demands from the Functional and Technical Design into demands for the specific implementation of this presentation layer and its different views.
- The implementation is made through a breakdown of the wanted functionality in components that realize that functionality.
  
The breakdown is based upon the separation of elementary functionality on a high technical level. Breaking down the total functionality in clearly divisible components and clearly describing the connections between those components a highly qualitative solution that consists out of highly qualitative sub solutions can be formed.
  
Per component input, processing and output is described. The connections between the components are described in an overview picture.

11.2.2 Presentation layer view definition

The presentation layer consists out of 4 views, these views are:

- FAB View
  Real-time monitoring of the business process takes place in this view.
- ICT View
  Real-time monitoring of the Web Services takes place in this view.
- Maintenance View
  The necessary data for the operating of the FAB View and ICT View can be generated and maintained in this view.
- Historical View
  Insight into the monitoring data of the past takes place in this view.

Remark:

Because these views differ a lot regarding the goal of the view and target group it is impossible to realize these views together. These views are therefore realized apart and the components these views are build upon are described in the next sub paragraph.

11.2.3 Presentation layer demands and wishes

Presentation layer demands and wishes:

- The realization of the dashboard tool is done with Microsoft .NET technology version 2.0 in the shape of ASP.NET 2.0 in combination with the Microsoft Atlas Framework.
- It must be possible to monitor the Web Services and the business process real-time, showed information must be updated real-time.
- It must be possible to correctly model a business process and the connections between that business process and the Web Services that implement that business process.
- The monitoring of a business process can be seen as viewing a traffic light, a business process runs well (performance is good, is available, 0 to little errors), runs not good and not bad (performance is not very good, not 100% availability and some errors occur) or runs bad (lousy performance, bad availability and lots of errors).
- It must be possible to monitor individual Web Services and there must be a clear distinction between performance, availability and errors within this individual monitoring.
- It must be possible to view monitoring data from the past.
11.2.4 Presentation layer components
The following components can be distinguished in the dashboard tool / presentation layer:

- Business Process Modeling Component
  This component contains the logic that is needed to:
  o Successfully interpret a BPEL4WS business process flow.
  o Define the connections between this business process and the Web Services through which
    this business process is implemented.

- Monitoring Rules Component
  This component contains:
  o The rule definition, the definition of monitoring rules.
  o The logic that is needed to connect monitoring rules to a business process definition and
    Web Services monitoring data.

- Database Connection Component
  This component is responsible for all the communication with the database in which the
  monitoring data is stored.

- Graphs / Tables Component and Traffic Light Component
  These components contain the techniques to visualize the monitoring data on Web Services level
  and business process level. The FAB View, ICT View and Historical View are build upon these
  components.

- Maintenance Component
  This component contains the interface in which the maintenance of the data needed for the
  operation of the FAB View, ICT View and Historical View can be done.

- Rule Interface Component
  This component contains the interface that enables a user to define rules that act as rules input
  for the Monitoring Rules Component.

Remark:
Above described components are described in more detail in the Technical Design document separately
supplied with this thesis paper.

11.2.5 Presentation layer overview
An overview of the total implementation of the presentation layer.
Figure 21 shows a component view of the presentation layer; the connections between the components and
the connections characteristics are shown.
Figure 22 shows a simplified input – output model of the dashboard tool / presentation layer. This picture
gives an overview of the main data streams that the presentation layer processes.
Figure 21: Component view of presentation layer

Figure 22: A simplified input – output model of the dashboard tool / presentation layer
11.3 Storage layer
The implementation of the storage layer is discussed; this paragraph is a short summary of the whole technical implementation described in the Technical Design document separately supplied with this thesis paper.

11.3.1 Approach
The approach that was followed during the making of the implementation for the storage layer was:
- Translate the demands from the Functional and Technical Design into demands for the implementation of the storage layer.
- Decide which demands and wishes influence the form of the implementation.
- The possible implementations and the factors that have influence on the implementation of the storage layer are described. The amount of factors that influenced the solution was so large that there was only one solution possible, a choice could not be made.

11.3.2 Storage layer demands and wishes
Storage layer demands and wishes:
- Standards must be used as much as possible on the different areas of implementation. The application of chain monitoring on different development and production environments touches different parties and users and thus standards must be used to ensure the best solution is developed.
- Implementation for both J2EE and .NET Web Services cannot be drastically different.
- Information from the past must be stored too, the past can be seen as not real-time, so today is real-time and yesterday and last week etc is the past.
- Storage of events per chain must take place as central as possible; this way a central source of information can be created. This source can then deliver all monitoring information needed.
- Storage in combination with Microsoft .NET technology must be realized in combination with MS SQL Server 2000, this is a common development standard in CC J2NET.
- It must be possible to easily validate message traffic from the DMZ to the LAN, this traffic goes through a number of fire walls and to make the validation easy the traffic must take place in the form of soap / http because this type of traffic can be directly validated by the fire wall without any extra effort made by the infrastructure.
- De central processing and storage of events near the storage server.
- Per day a summarization job needs to run over the monitoring data of yesterday. This data needs to be summarized and stored so that it is available to be viewed.

11.3.3 Storage layer implementation
A description of the decisions made to come to the chosen implementation of the storage layer. The demand of traffic message validation between the DMZ environment and the LAN environment eliminates the possibility of a direct connection between the Web Server on which the events are generated and the database to store those events in. As a result a Web Service is needed to store the events in the database.
The chosen implementation of the storage layer thus consists of 3 parts + the historical part:
- Storage of events
- Processing of events: .NET
- Processing of events: J2EE
- Historical storage of events

For each of these implementation parts the demands and wishes are put together and on the basis of these demands and wishes the best implementation is formed. The result of the first 3 implementations is shown in Figure 23 and 24.
The historical storage of events implementation consists of a MS SQL Stored Procedure that is run daily and summarizes the monitoring data of the day before.
Remark:
For a full description of the above described decision and the implementation parts see the whole technical implementation described in the Technical Design document separately supplied with this thesis paper.
Figure 23: Technical implementation of the processing and storage of events for .NET Web Services

Figure 24: Technical implementation of the processing and storage of events for J2EE Web Services
12. Event-monitoring: Realization

A description of the realization of event-monitoring, experiences with this realization and an evaluation with the target group is discussed here.

12.1 Experiences

The experiences of realizing this iteration are discussed via a number of Proof of Concepts. These Proof of Concepts give an overview of the lessons learned and techniques used to realize the concepts stated in the Technical Design. To give an idea of how the monitoring of Web Services and a business process will look like some screen shots are included and shortly described.

12.1.1 List of Proof of Concepts

The following list of Proof of Concepts is related to the realization of event-monitoring:

- The making of a .NET Windows Service that listens to WMI via a thread implementation.
- The making of a .NET Web Service
- The making of a client proxy to a .NET Web Service in a .NET Windows Service
- The making of a J2EE Custom Service
- The making of a J2EE Web Service
- The making of a client proxy to a .NET Web Service in a J2EE Web Service
- The making of a T-SQL stored procedures for the summarization of monitoring data of yesterday.
- The making of an ASP.NET 2.0 Web Site that shows graphs and tables.
- The making of an ASP.NET 2.0 Web Site that shows a traffic light.

12.1.2 Screen shots

Some screen shots are shown in this section to give an idea of the realized products of the iteration event-monitoring. The screen shots are taken while monitoring a chain of 2 Web Services; these Web Services were equipped with event-generation and event-collection. The screen shots show a concept version of the FAB and ICT View, the functionality offered and described in the Functional and Technical Design is completely realized, the layout of these View’s is however in a concept phase.

The following screen shots concern the monitoring of a chain of Web Services or business process via the FAB View.

Figure 25: Screen shot of the rule interface of the FAB View, users can fill in the rules that determine the health of a business process (activity) or Web Service shown via the traffic light interface.
Chain monitoring for Web Services

FAB View:

1. Regelinterface

2. Verkeerslicht

Figure 26: Screen shot of the traffic light interface of the FAB View, users can monitor a business process (activity) or Web Service through the definition of rules and the real-time checking of these rules onto monitoring data and the resulting traffic light indication of the health.

The following screen shots concern the monitoring of individual Web Services via the ICT View.

Figure 27: Screen shot of the availability section in the ICT View, per Web Service the availability on hour level can be monitored.
Figure 28: Screen shot of the performance section in the ICT View, per Web Service method the performance of individual requests on hour level can be monitored.

Figure 29: Screen shot of the performance section in the ICT View, per Web Service method the performance on hour level can be monitored via a pie-like view.
12.2 Evaluation

An evaluation of the results of this iteration. For a description of the total evaluation see Chapter 2.6.3.

12.2.1 Approach

A review of the results of this iteration a verification of the implementation and realization (Functional, Technical Design and Realization) against the chosen approach or solution stated in Chapter 2.5 and 8. The approach for this verification is as follows:

- Find the suitable persons for the evaluation of this iteration.
- Discuss the results of these iterations with them, through discussing the following list of items:
  o Description of the iteration event-monitoring.
  o Demands and wishes for the implementation of event-monitoring.
  o The elaboration of the concepts into the Functional Design.
  o The elaboration of the concepts into the Technical Design.
- Through asking questions about the results of the implementation and realization and the relationship between these results and the chosen approach or solution the evaluation of this iteration is done. Besides these questions a large part of the evaluation consists of a little bit of self reflection.

12.2.2 Evaluation results

The results of the evaluation done for the third iteration event-monitoring.

Evaluation persons

The evaluation of this iteration is done with the following persons:

- ICT represented by:
  o Saber Karmous
  Saber Karmous is a .NET architect within the CC J2NET; he has a lot of experience with .NET and is associated as an architect with a lot of projects that have delivered .NET Web Services within ICT. Next to his part in the evaluation of this graduation project Saber is associated with this graduation project as a technical assistant for .NET.

- FAB represented by:
  o Ronald van der Lee
  o Ronald van Dijk
  Both are Functional Application Managers for the business line ‘Schade’. They are responsible for the daily operation of a number of applications that are used within this business line. Both are associated with this gradation project as functional assistants for the realization of the FAB View.

Evaluation results

The material that is discussed during this evaluation can be found in the chapters 8, 9, 10 and 11.

Verification

The end results of the verification of the 3rd iteration.

Approach

The separation of this iteration with the previous iterations, together with the way of managing requirements have created a clear view of the demands and wishes that form the basis of this iteration. Prince2 has helped to manage the progress of this iteration and define the deliverables, part of this management can be found in the Functional and Technical Design. The heritage of the previous iterations had to be dealt with; the possibilities were limited in that way. Separation of the iterations event-collection and event-monitoring was hard, collecting events and storing them is linked together but cannot be seen as one functional item. Storage of events is definitely connected to the presentation of those events; the information for the presentation layer has to come out of the storage layer.
To distinguish functionalities from techniques was also a bit of a difficulty in this iteration, both items overlap and a clear distinction between those items cannot be made easily.

This iteration was especially different from the previous iteration in the number of target groups that had to be dealt with. FAB and ICT (which is divided in an infrastructure part and a software part) formed 3 target groups that had different demands and wishes and overlapped regarding interests in both presentation and storage layer. Mentality, ways of thinking, starting points differ and must be taken into account while communicating with a target group and achieving a good end result.

The combining of demands and wishes from the different target groups was done without major difficulties, prioritizing the demands and wishes was more difficult because the interests of the different groups were far apart.

Implementation and realization
The separation of presentation layer and storage layer could not be clearly concluded out of the initial demands and wishes for this iteration, to make a clearly supported guideline for this separation was hard and might not have been done that well. Through describing the approach the demands and wishes more easily became clear and a good basis for the Functional and Technical Design was created.

The breakdown into components for both the presentation and the storage layer has resulted in a clearly separated collection of demands and wishes. Functionality needed for the two layers could be implemented and realized in a generic and feasible way because of this clear separation of demands and wishes.

The separation into two layers and the breakdown into components have made the evaluation of the different components a lot easier because the single components could be evaluated separately. Demands and wishes could be translated easier towards component demands and wishes. Unit testing and a good integration can be done easier and in an earlier phase of the development of components when the interfaces between components are described and defined in an early stage. The interfaces take care of the fact that the data flow throughout the components is well defined and clearly described. The definition of interfaces in the presentation layer has taken a lot of time but has succeeded.

The way of realizing through a number of Proof of Concepts has helped to evaluate this iteration in an early stage. The functionality offered by the presentation and storage layer could be showed through prototypes that were realized in the Proof of Concepts, evaluation between target groups and software developer could start before the deliverables of the iteration were finished.

Results / deliverables
The presentation layer deliverables are the FAB View and ICT View, these views form the main deliverables of the presentation layer, the other views are left out here because of the small amount of functionality they offer compared to these views.

The FAB View has achieved its goals, it allows for the monitoring of chains of Web Services or a business process that is implemented by a number of Web Services. Through the traffic light interface it is possible to continuously monitor a business process.

The ICT View is well suited for the real-time monitoring of a Web Service, the representation of monitoring data via graphs and tables is well suited for the providing of information about the health of Web Services.

The storage layer deliverables are the processing and storage agent implementations and the data model for the database. The data model for the database is robust and easy to scale. The processing and storage agents offer flexibility and are realized in a generic and feasible way.
Part 4:
Conclusion and Future Work
13. Validation / evaluation

An evaluation of the whole thesis paper is done here. For a description of the total evaluation see Chapter 2.6.3.

13.1 Approach of the validation

The verification of the 1st, 2nd and 3rd iteration is done in the realization section of each iteration. The validation of this graduation project is not yet done. A review of the results of this thesis paper is a validation of the problem definition (Chapter 1) against the found concepts in the literature study (Chapter 3, 4 and 8).

The approach for this validation is as follows:

- Find the suitable persons for the evaluation of this graduation project.
- Describe the detailed problem definition in terms of the iterations chosen in the approach.
- Describe the results of the literature study, the chosen concepts per iteration.
- Through asking questions about the results of literature study and the relationship between these results and the problem definition the validation of this graduation project is done. Besides these questions a large part of the evaluation consists of a little bit of self reflection.

13.2 Evaluation results

The results of the evaluation done for the whole graduation project.

Evaluation persons

The evaluation of the whole graduation project is done with the persons that have done the evaluation of the 1st and 2nd iteration and the 3rd iteration:

- ICT represented by:
  - Duncan Doyle
  - Saber Karmous
- FAB represented by:
  - Ronald van der Lee
  - Ronald van Dijk

Evaluation material

The material that is discussed during this evaluation can be found in the chapters 1, 3, 4 and 8.

Validation

The end results of the validation of this graduation project.

Approach

The initial problem definition with which this graduation project is started is very short. Define the problem definition in more detail through the description of important factors made the problem definition a lot more understandable.

A clear and detailed definition of the problem is a good way to make a detailed comparison between the chosen concepts and the problem definition, parts of the concepts can be linked to parts of the problem definition and a direct validation of the problem definition against the chosen concepts can be made which results in a detailed validation.

It is however difficult to get all the factors that influenced the problem definition clear, mainly because of the complex organization structure (several staff services, business lines, etc...). The problem definition could not be made clear straight from the beginning which is not good for the validation, a clear distinction between the problem definition and proceeding iterations contributes to a more direct validation between problem definition and chosen concepts.

The way of describing the concepts through an introduction, application onto the current iteration, describing the latest conditions, impossibilities versus possibilities and an evaluation provides a basis for making a well thought decision about which concept to chose.

Describing concepts on a high abstraction level, not going into technical details takes care of the fact that the concepts can easily be compared to the problem definition and initial demands and wishes which are also
described on a high abstraction level. Finding the most suitable concept to solve a problem is more easily when both are described on the same abstraction level. The definition of possibilities and impossibilities isn’t always done in a clear transparent way; technical detailed knowledge is used to define some of the possibilities and impossibilities. To keep the abstraction level of the concepts description high concessions are done to the description and understandability of the possibilities and impossibilities.

Results / deliverables

By clearly defining the influence of the factors that have influenced the problem definition the final detailed problem definition is build in a transparent way. The problem definition is detailed but is it complete? It is hard to check if all the factors that have influenced the problem definition are described. The found concepts in the literature studies are divers, the chosen concepts match the needs defined in the problem definition looking at the results of the evaluations. Multiple concepts that fulfill a part of the problem definition in a similar way are not found, a difficult decision to choose a concept is not made. This raises the question if enough concepts are found to speak about a best concept to fulfill the needs described in a part of the problem definition.

To see if the problem definition satisfies the chosen concepts of the literature study an overview of the relations between problem definition factors and chosen literature study concepts is made.

- Event-generation and event-collection
  The part of the problem definition that applies on the iterations event-generation and event-collection is mainly concerned with:
  - The development of new Web Services and the existence of a number of Web Services.
  - The SOA architecture that is responsible for the forming of chains of Web Services, a large number of Web Services that all realize different functionality.
  - The difference between Back Office and Front Office that is responsible for a division in the infrastructure and run-time environment of these Web Services.
  The chosen concepts in the literature study are:
  - Aspect Oriented Programming
    AOP is a technique that can be used to physically separate the implementation of chain monitoring from the Web Services and thus make an implementation of chain monitoring easily available to implement on both existing and new Web Services.
  - Java Management eXtensions and Windows Management Instrumentation
    JMX and WMI are implementations of architectures that enable the monitoring (generation and collection of events) of applications like Web Services. These implementations are the standard for .NET and J2EE run-time environments and:
    - These standards can handle large amounts of monitoring information.
    - These standards cover the Front and Back Office run-time environments of Web Services within ICT.

  The chosen literature study concepts thus cover the concerned parts of the problem definition.

- Event-monitoring
  The part of the problem definition that applies on the iteration event-monitoring is mainly concerned with:
  - ICT is not the only party interested in chain monitoring for Web Services, customers of ICT have similar interests in chain monitoring for Web Services.
  The event-monitoring literature study showed that there are the following needs:
  - FAB wishes to view the health of a business process realized by a chain of Web Services.
  - ICT wishes to view the health of an individual Web Service that is part of such a chain of Web Services.
  The chosen literature study concept Web Services Composition makes a connection between a business process and the Web Services realizing that business process. Web Services Composition consists of a modeling language in which a business process can be modeled in a way that the connections with Web Services (realizing certain bits of functionality of that business process) can also be modeled.
14. **Conclusions and Future Work**
The conclusion of this thesis paper and future work that can be done after this thesis paper.

14.1 **Conclusion**
An overview of the total realization of the iterations and the experiences while realizing the different iterations. The realization discussed in Chapter 7, 12 and 13 is part of this summary.

14.1.1 **Evaluation results**
Evaluations have taken place at the end of the 1st and 2nd iteration, after the 3rd iteration and a total evaluation is done in the conclusion.
The results of the verification done in Chapter 7 and 12 are that the products that are realized in the 3 iterations event-generation, event-collection and event-monitoring matches the approach or solution chosen in Chapter 2.5.
The results of the validation done in Chapter 13.1 are that the chosen literature study concepts match the aspects discussed in the detailed problem definition.
Most of the evaluation results are positive; some self criticism about the approach but conversations with both FAB and ICT left a positive impression of the whole approach and graduation project behind.

14.1.2 **Realization deliverables**
The realization deliverables are of course the deliverables of the different iterations.
The deliverables are:
- A total package of software and documentation to equip a .NET and / or J2EE Web Service with event-generation.
- A total package of software and documentation to enhance the run-time environment (IIS and WMI, IBM Websphere and JMX) for the collection of events generated by .NET and J2EE Web Services.
- A dashboard tool (+ documentation) that consists out of the following views:
  - FAB View
  - ICT View
  - Historical View
  - Maintenance View

These views together realize:
- The real-time monitoring of a business process or chain of Web Services
- The real-time monitoring of individual Web Services
- The maintenance needed for the other 3 views
- The viewing of monitoring information about (chains of) Web Services from the past.

14.1.3 **Conclusion**
The evaluation results are all positive, conversations with FAB and ICT proved that a product has been realized that fulfills their needs and is realized via an approach that has succeeded.
The iterative approach chosen in Chapter 2 has resulted in a well structured solution in which almost all the demands and wishes of the different target groups are met and some useful and almost production ready deliverables are delivered.
The end result of this graduation project is a product that shows that the monitoring of chains of Web Services with this approach and this deliverables can be done; the initial problem definition can be solved within the organization of Fortis with this solution.
The chosen approach or action plan has resulted in a product that solves the stated problem definition for Fortis Insurance Netherlands.
The chosen approach, action plan and requirements engineering and management are not solely connected to Fortis Insurances Netherlands, the chosen concepts stated in the literature study are not specifically pointed to the situation at Fortis, implementation and realization of these concepts that is done in the Functional and Technical Design is however pointed at Fortis.

The iterative approach and way of engineering and managing requirements are the guideline and basis of the solution described in this thesis paper. The approach and way of handling requirements have proven that an application of these ways of working onto an organization like Fortis can have a good end result. A similar way of working with a similar problem definition and similar literature study concepts applied upon a different organization and infrastructure is possible, the approach and problem definition can form the basis for such an approach and a similar result can be obtained.

14.2 Future work

The future work for this graduation project can be divided into 3 areas, namely the application of the approach, concepts and deliverables of this graduation project on a broader scale. The approach, concepts and deliverables of this graduation project form the main part of this thesis paper.

All of these categories are somehow narrowed down for this graduation project but can be applied on a broader scale in the future. These three categories will be discussed in this paragraph to give an idea of future work that can be done after this graduation project.

14.2.1 Application of graduation project concepts

The concepts chosen in the two literature studies are concepts that have been applied to (chains of) Web Services in this graduation project. Chain monitoring can be interpreted in a number of ways, chain monitoring for Web Services as defined in this graduation project is one of those interpretations.

Web Services or chains of Web Services and their performance, availability and errors are just the top of the ice berg, a Web Service runs in a run-time environment, a run-time environment runs on an operating system, and an operating system runs on hardware. All these factors have an important role in the performance, availability and errors of chains of Web Services. Information about these factors can definitely be helpful while monitoring chains of Web Services and can define the monitoring of chains of Web Services in a much broader sense, can give the term chain monitoring a bigger meaning.

Web Services and chains of Web Services realize elementary pieces of functionality, while realizing this functionality communication with databases, other software applications, etc… is part of the operation of a Web Service. These pieces of the chain are not contained in the definition of chain monitoring for this graduation project but do matter for the monitoring of a chain. Gathering information about the performance, availability and errors of these other chain pieces can give a lot of information about the health of these pieces, chains of these pieces can be separately monitored in a form of chain monitoring for software applications for instance.

The concepts discussed in the literature study cannot only be applied to chains of Web Services but also to other pieces of that chain of Web Services that are not yet monitored. Zooming in on Web Services, Web Services are the top of the ice berg, run-time environments and operating systems have a major influence on the operation of Web Services and monitoring information of those environments can thus be of great help while monitoring Web Services.

14.2.2 Application of graduation project approach

The approach chosen in this graduation project consists of the iterative approach, the way of engineering and managing requirements, the way to describe and choose concepts and the elaboration of concepts in the Functional and Technical Design. The approach can achieve good results as shown in this graduation project but is not specifically designed for this problem or the organization and infrastructure of Fortis Insurances Netherlands. An application of this approach can be done onto a different problem within Fortis or different or similar problems outside Fortis.

The approach takes care of the way deliverables are obtained, or the way a concept is described. The content of a project, the problem definition or found concepts is not closely related to an approach, an application of this approach (and possibly a better elaboration or refinement of the approach) can take place onto another problem, another organization or another infrastructure.
14.2.3 Application of graduation project deliverables

The graduation project deliverables mentioned in Chapter 14.1.2 are in a concept phase, this means that the functionality is realized completely, the layout of that functionality in terms of source code and web site layout is not yet completed.

Next to the fact that the realized deliverables are not yet in a production state the effects of for instance the appliance of event-generation on the operation of a Web Service is not yet fully discovered. More load tests with other more specific scenarios can help to discover the overhead of event-generation on both Web Services and run-time environments.

Not all demands and wishes are fulfilled, the demand to be able to monitor on traffic message level is not fulfilled because the resources for this demand were not available and it was impossible to realize a generic solution for Web Services within this graduation project because of the amount of concerned target groups.

Future work to apply the graduation project deliverables on a number of (chains of) Web Services consists out of finalizing the deliverables, perhaps fulfill the few not fulfilled demands and further investigate the overhead of event-generation (AOP).
Bibliography

This section contains the references to external sources of information, external means outside of this graduation project.

Introduction (Chapter 1):
1. IBM InfoCenter Terminology: Definition of ‘Web Services’
2. Lucent Glossary – W
   On-line at: www.lucent.com/search/glossary/w-definitions.html
3. A Word definition from Webopedia: What is SOAP?
4. A Word definition from Wikipedia: Service Oriented Architecture

Approach and ‘Event-generation and event-collection: Preparation’ (Chapter 2, 3):
6. Interview with Eddy van der Heiden, Senior Functional Developer, Fortis Insurances Netherlands. Subject: Content and shape of signals from (web) services with the business as point of view, date of interview: 2005-11-17.
7. Interview with Martijn Drost, Technical Specialist Infrastructure, Fortis Insurances Netherlands-ICT. Subject: JMX within Fortis ASR, date of interview: 02-12-2005
8. Interview with Leendert Versluijs, Technical Specialist Infrastructure, Fortis Insurances Netherlands-ICT. Subject: WMI within Fortis ASR, date of interview: 02-12-2005
10. Interview with Evert Heijning, Senior Technical Specialist Infrastructure, Fortis Insurances Netherlands-ICT. Subject: Project ‘End-to-end monitoring’, date of interview: 10-10-2005

Event-generation and event-collection: Literature study (Chapter 4):
Web Service Offerings Language:
Aspect Oriented Programming:
   On-line at: http://portal.acm.org/citation.cfm?doid=383845.383853

Java Management eXtenstion:
    ISBN: 0-596-00245-9, 312 pages
19. Java Management eXtension (JMX) website at Sun Microsystems

Windows Management Instrumentation:
22. MSDN Library – WMI Reference

Event-monitoring: Preparation (Chapter 8):

Event-monitoring: Literature study (Chapter 9):
Basis:
25. Word definition from Webopedia: What is business process?
26. A Word definition from Wikipedia: business process:

Web Services Composition:

Workflow Management and Web Services:


35. Workflow Management Coalition, On-line at: http://www.wfmc.org

Business Process Management:


Evaluation:


List of terms:
Appendices
This chapter describes the appendices used for this thesis paper.

Appendix A: List of Terms
This appendix describes terms that are not discussed or not very well discussed during the thesis paper.

Aspect Oriented Programming (AOP)
Aspect oriented programming is an extension to the well known ways of for instance Object Oriented programming. It is possible via AOP to insert a piece of source code A into a piece of source code B without a visual reference of A within B. Considering the structure of this extension this concept fits the best with Object Oriented Programming.

DMZ
This is the so-called ‘demilitarized zone’ of Fortis ASR in which Web Services are hosted; this environment is separated from the internal network and the internet through a number of firewalls.

LAN
This is the internal network of Fortis ASR in which Web Services are hosted; this environment is separated from the DMZ and the internet through a number of firewalls.

IBM Websphere
IBM WebSphere is the portal platform of IBM, based on the J2EE standard. The WebSphere platform offers a collection of technologies and products, among which are IBM WebSphere Portal Server (WPS) and WebSphere Application Server (WAS), pointed towards the development of robust and scalable portal solutions.

Integrated Development Environment (IDE)
And Integrated Development Environment or IDE is computer software that supports a software developer by the development of computer software.

Internet Information Services (IIS)
IIS (Microsoft Internet Information Services or Server) is a collection of server services for the internet, specially meant for computers that run Windows for Operating System. A computer with IIS then becomes a so called web server.

Java Management Extensions
The Java Management Extensions (JMX) technology is an open technology for management and monitoring that can be deployed wherever management and monitoring are needed. By design, this standard is suitable for adapting legacy systems, implementing new management and monitoring solutions and plugging into those of the future. [42]

Namespace
A namespace is a unique identifier / location for an information source / provider within WMI / JMX. Thanks to this namespace information can be located in WMI / JMX.

Web Services Composition Language
A modeling language to realize the concept of Web Services Composition. A modeling language to combine the functionality of multiple Web Services into a single combined functionality.

Windows Management Instrumentation
WMI, an API in the Windows operating system that enables devices and systems in a network, typically enterprise networks, to be managed and controlled. Utilizing the Common Information Model (an industry standard for describing data about applications and devices), WMI allows network administrators to query and set information on workstations, applications and networks.
Appendix B: List of Figures

This appendix describes a list of figures that is used throughout the whole thesis paper.

- Figure 1: Organization of ‘Fortis Insurances Netherlands’.
- Figure 2: SOFIA layers, functionality and applications.
- Figure 3: The relationship between validation and verification
- Figure 4: Differentiation of levels on which signals can be monitored.
- Figure 5: Partial UML class diagram of the WSOL concepts. [11]
- Figure 6: Simplified architecture of JMX.
- Figure 7: Simplified architecture of WMI.
- Figure 8: decentralized collection bin per platform.
- Figure 9: centralized collection bin for all platforms.
- Figure 10: Event-generation, event-collection en event-monitoring in 1 glance.
- Figure 11: Signal implementation
- Figure 12: Implementation of event-generation
- Figure 13: Overview of web services technology [27]
- Figure 14: Reference model of the Workflow Management Coalition (©WfMC). [34]
- Figure 15: An example of a BPMa architecture. [39]
- Figure 16: Global data model of the presentation layer
- Figure 17: BPEL4WS data model
- Figure 18: Processing centrally
- Figure 19: Processing de centrally
- Figure 20: Data model of storage layer
- Figure 21: Component view of presentation layer
- Figure 22: A simplified input – output model of the dashboard tool / presentation layer
- Figure 23: Technical implementation of the processing and storage of events for .NET Web Services
- Figure 24: Technical implementation of the processing and storage of events for J2EE Web Services
- Figure 25: Screen shot of the rule interface of the FAB View, users can fill in the rules that determine the health of a business process (activity) or Web Service shown via the traffic light interface.
- Figure 26: Screen shot of the traffic light interface of the FAB View, users can monitor a business process (activity) or Web Service through the definition of rules and the real-time checking of these rules onto monitoring data and the resulting traffic light indication of the health.
- Figure 27: Screen shot of the availability section in the ICT View, per Web Service the availability on hour level can be monitored.
- Figure 28: Screen shot of the performance section in the ICT View, per Web Service method the performance of individual requests on hour level can be monitored.
- Figure 29: Screen shot of the performance section in the ICT View, per Web Service method the performance on hour level can be monitored via a pie-like view.
Appendix C: List of Tables

This appendix describes a list of tables that is used throughout the whole thesis paper.

- Table 1: Advantages and disadvantages of a new and existing implementation of event-collection.
- Table 2: Advantages and disadvantages of event-collection concepts.
- Table 3: Description of factors of importance to the choice of an AOP tool.
- Table 4: Comparison of AOP tools for J2EE.
- Table 5: Comparison of AOP tools for .NET.
- Table 6: An overview of signals, signal levels and signal information.
- Table 7: Signal to management interface mapping for JMX.
- Table 8: Signal to management interface mapping for WMI.
- Table 9: Functional documentation extensions needed for the implementation of event-generation and event-collection.
- Table 10: Signal implementation for JMX.
- Table 11: Signal implementation for WMI.
- Table 12: Technical signal implementation.
- Table 13: Description of factors that are of importance to the choice of a modeling language.
- Table 14: Comparison of possible modeling languages.
- Table 15: Advantages and disadvantages of concepts for the implementation of the storage layer of event-monitoring.
- Table 16: Technical signal implementation used for the summarizing of monitoring data.
Appendix D: List of other documents
This appendix describes a list of documents that are used to make this thesis paper. Both name and a short summary of the contents of the documents are supplied.

- Prince2 Documentation graduation project
  These documents can be found on the enclosed CD-Rom.
  - Project Proposal: Chain monitoring for Web Services
    This document describes the initiation and basis of this project.
  - Project Approach: Chain monitoring for Web Services
    This document describes the approach taken to implement and realize this project.
  - Phase approach: Event-generation
    This document describes the approach taken to implement and realize this phase.
  - Phase approach: Event-collection
    This document describes the approach taken to implement and realize this phase.
  - Phase approach: Event-monitoring
    This document describes the approach taken to implement and realize this phase.

- 1st and 2nd iteration: Event-generation and event-collection
  These documents are either separately supplied with this thesis paper in a document called “Appendices: Iteratie 1 & 2 : Event-generatie en Event-verzameling” or can be found on the enclosed CD-Rom.
  - Literature study 1st and 2nd iteration
    This document contains the complete literature study of the 1st and 2nd iteration event-generation and event-collection. The literature study preparation, concepts and conclusion are described in this document.
  - Functional Design 1st and 2nd iteration
    This document contains the Functional Design of event-generation and event-collection; the concepts AOP, JMX + WMI are described and defined on a functional level.
  - Technical Design 1st and 2nd iteration
    This document contains the Technical Design of event-generation and event-collection; the concepts AOP, JMX + WMI are described and defined on a technical level.
  - Evaluation 1st and 2nd iteration
    This document contains the evaluation of event-generation and event-collection done with ICT architects, verification and validation of the problem definition, approach, literature study and realization is described in this document.
  - Load test analysis
    This document contains the analysis, results and conclusion of a load test done to investigate the overhead of event-generation on the normal operation of a Web Service.

- 3rd iteration: Event-monitoring
  These documents are either separately supplied with this thesis paper in a document called “Appendices: Iteratie 3 : Event-monitoring” or can be found on the enclosed CD-Rom.
  - Literature study 3rd iteration
    This document contains the complete literature study of the 3rd iteration event-monitoring. The literature study preparation, concepts and conclusion are described in this document.
  - Functional Design 3rd iteration
    This document contains the Functional Design of event-monitoring, the concept WSC and the presentation and storage layer are described and defined on a functional level.
  - Technical Design 3rd iteration
    This document contains the Technical Design of event-monitoring, the concept WSC and the presentation and storage layer are described and defined on a technical level.
  - Evaluation 3rd iteration
    This document contains the evaluation of event-monitoring done with ICT architects and FAB, verification and validation of the problem definition, approach, literature study and realization is described in this document.