Towards an architectural framework for IT-enabled, continuous auditing at health insurers

Kennes, J.H.J.

Award date:
2008

TECHNISCHE UNIVERSITEIT EINDHOVEN
Department of Mathematics and Computing Science

MASTER'S THESIS

Towards an Architectural Framework for IT-enabled, Continuous Auditing at Health Insurers

by

J.H.J. Kennes

Supervisors:

prof.dr.ir. P.W.P.J. Grefen (TU/e)
dr. N. Sidorova (TU/e)
dr. A.T.M. Aerts (TU/e)
W.A. Braal RE (PricewaterhouseCoopers)
drs. R.J.W.P. Bastiaansen (PricewaterhouseCoopers)

Eindhoven, November 2007

Management summary

The accountancy world is changing. Tightening regulations and rapid information technology developments awaken accountants to the opportunities of IT. Some accountants believe that IT can solve anything, and that automated and continuous auditing is only a matter of time. Others think that IT could never take over the work of accountants and they strongly resist to adopting new IT solutions. The truth is somewhere in middle.

There is still a lack of clarity and uncertainty on IT-enabled (automated) and continuous auditing. Questions are raised on the feasibility of IT-enabled auditing, on how IT-enabled auditing should look, on which technology to use for IT-enabled auditing, and on how to assign responsibilities with IT-enabled auditing. Could IT-enabled auditing lead to continuous auditing instead of performing the audit once a year, is also questioned. This leads to the following research question:

In what way can PwC, by means of IT, automate its audit process and provide new services for the health insurer industry, with a view to provide continuous assurance as defined by long-term strategy?

A conceptual design, supported by an architectural framework of IT-enabled, continuous auditing at health insurers gives answers. Within the health insurer, this research focuses on auditing the premium process. In the future, the design can be extended to different processes within the health insurer.

The audit process

Accountants provide independent opinions on different subject matters, e.g. financial reporting and compliance. The audit consists of two parts: the interim audit and the final audit. During the interim audit, systems and processes are investigated to address the level of internal control. With the internal control process, organizations provide assurance on achieving goals concerning effectiveness and efficiency of operations, reliability of financial reporting, and compliance with laws and regulations. During the final audit, the financial statements are checked and analyzed. An audit is based on the internal control of an organization, see Figure 1. Dependent on the level of internal control, the financial statements are analyzed more or less extensively. Internal control on operational level is implemented by application (automated) controls and manual controls. For controls to function correctly, the depend on general IT controls and the control environment.

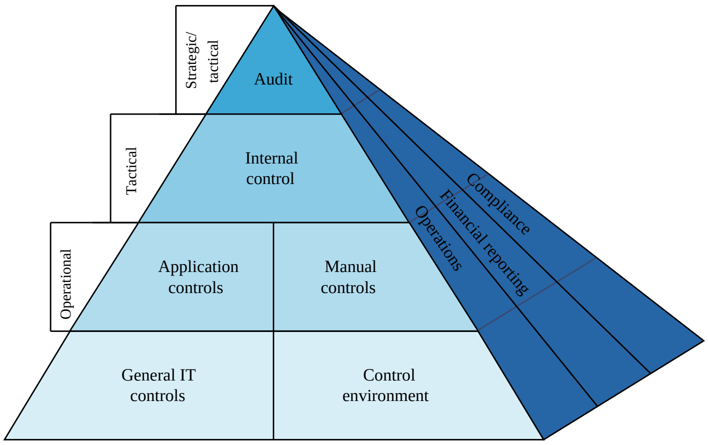

Figure 1: Framework of audit dependencies

The architectural framework

An architectural framework provides a structured description of a system by its components, their relationships to each other and to the environment, and its design, and relates various viewpoints and associated modeling techniques of the structured description. A viewpoint gives a specification on how to construct and use views. Views are a representations of the system from a specific perspective. For the architectural framework used in this research, four viewpoints are distinguished: a *process* viewpoint, a *data* viewpoint, an *organization* viewpoint and a *system* viewpoint.

The audit system for IT-enabled, continuous auditing

As a starting point for designing the audit system for IT-enabled, continuous auditing, or simply the audit system, a point in time ten years from now is taken. This is to be independent of current technical limitations.

Functional description and process viewpoint

The functional description of the audit system distinguishes three levels: a strategic level, a tactical level, and an operational level. At the strategic level, an agreement on collaboration between the health insurer and the accountant is established. At the tactical level decisions for the execution of the collaboration are made and communicated to the operational level. At the operational level, the health insurer sends automatically and digitally data from its premium process to the operational audit system. The audit systems checks the data, forms an opinion, and communicates the opinion to the health insurer.

The audit system is designed to be flexible. Changes in laws and regulations, new input data formats from the health insurer, different checks on data, and various output reports with findings are issues the audit system needs to deal with. Since an audit is based on the level of internal control of an organization, this is also taken into consideration by the audit system. Flexibility is achieved through the use of a configuration. The input and output format, as well as the checks to be done, are stored. The configuration is filled in at the tactical level, considering requirements from the health insurer, from laws and regulations, and the level of internal control of the health insurer.

At the operational level, operational data (e.g. modifications of insurants) and process data (in the form of event log files) are received by the audit system. After a conversion to the internal format of the audit system, the data is checked by functional modules. A few modules are already defined, e.g. check access to applications, check exception processing of the premium process, and check data flow between applications. The modules can be extended in the future. Checks by the functional modules result in observations. All observations are combined, and a report with findings and an opinion is sent to the health insurer.

Data viewpoint

This research provides a data viewpoint of different types of data used and produced by the audit system. Of great importance for the audit system is the configuration. The configuration contains input settings, output settings and checked processes. Input settings are e.g. which population of data should be subject of the audit, output settings include after which period a report should be generated. A checked process consists of a process with one applied check. Not all processes need to be checked, dependent of the level of internal control. A check consists of one or more rules, e.g. compliance rules or assertions. The rules are also stored in the configuration.

The opinion provided by accountants when performing an audit, is in the audit system captured by a certificate. This certificated is specially designed for the audit system and differs from traditional auditors' certificates. The certificate consists of a header, a footer, and a body. The header contains information on the audit assignment, e.g. the company that is audited. The footer contains an electronic certificate that verifies the accountant's identity. The electronic certificate replaces the accountant's signature in traditional certificates. The body of the certificate that is provided by the audit system, contains a general opinion and can be extended with clauses containing opinions on the checked rules. The processes that are checked, can also be included.

Organization viewpoint

With the introduction of the audit system, responsibilities of involved parties will change. The most important change is that accountants become responsible for the operational audit system. Although system experts are probably involved, accountants strongly resist to this responsibility. Their lack of knowledge, a different focus of their education, and the stiffness of the audit profession make the introduction of the audit system difficult.

Another change in responsibilities can be found at the operational level, where the health insurer is responsible for storing data and events in the right format and location. The accountant is responsible for the communication of operational data and process data to the audit system.

System viewpoint

The system design of the audit system is achieved by using architectural patterns. The audit system takes data, performs checks on data, and delivers data, the *pipe and filter* pattern is therefore applied. A configuration can change regularly, for modeling the use of a configuration by the audit system, the *reflection* pattern is appropriate. The functional modules are seen as services, and have to be extendable. The *client-server* pattern is used here: a client communicates to one or more servers (services) and keeps track of the available servers. Not only is this useful for functional modules, also converting data and reporting can be modeled similar. Different services for changing input formats or reporting formats can be added in the future.

For the implementation of the patterns, a service oriented approach is chosen. Functionalities within the system are all seen as services and data can be communicated internally, between services, and externally by XML. For the implementation of functional modules, a rule-based engine is suitable. Domain-specific rules are examined to proof the satisfaction of domain-specific facts.

Conclusion

Is IT-enabled, continuous auditing feasible? Technically speaking, the question is yes. A conceptual design is provided for IT-enabled, continuous auditing of the premium process at a health insurer. Technologies that can be used for implementing the audit system, are already available. However, risks that might stand a future implementation in the way, are the complexity of eliciting domain-specific rules for the functional modules, the involvement of a third party to issue electronic certificates for the identification of the accountant, the co-operation of the health insurer, and the resistance of accountants internally. Further research into these risks and a more detailed specification of the audit system are needed before the audit system can be implemented.

The audit system is specially designed to audit the premium process of the health insurer. Since most processes of health insurers are automated and resemble to the premium process, a extension to these automated processes of health insurers is feasible. Automation is a precondition for the audit system, which is possible due to the simple routine operations and standardized work processes at health insurers. Other organizations with similar characteristics qualify for an IT-enabled, continuous audit as well.

In the beginning of this summary, an overview was given of the audit process and its dependencies on internal control, see Figure 1. When performing audits continuously using the audit system, the audit becomes partly an operational process and shifts within the framework to the operational level, see Figure 2. The correct functioning of the audit system depends on the configuration of the audit system. The results of the audit are communicated to the management, and the system configuration is established at the tactical level, by the “audit control”.

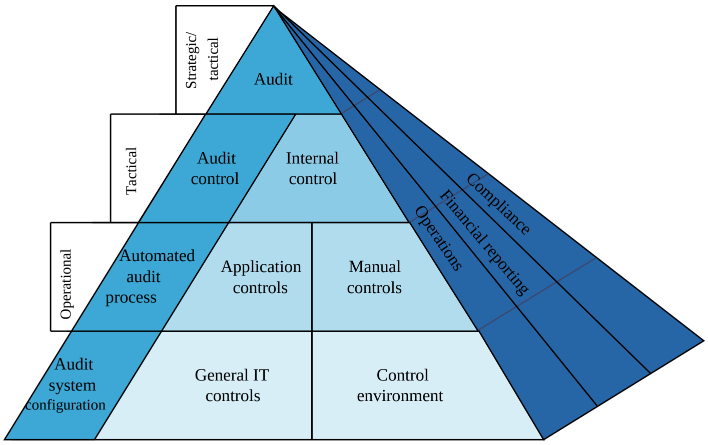

Figure 2: Shift of audit in framework of audit dependencies

In this research, new possibilities for IT-enabled, continuous auditing are investigated. Since IT-enabled, continuous auditing seems to be feasible, the next step will hopefully be to implement such a system and gain experience with IT-enabled, continuous auditing. A basis for an implementation is provided by this research.

Preface

When I started to realize it was time for me to leave an exuberant and enjoyable university life behind, PricewaterhouseCoopers gave me the opportunity to do an interesting graduation project with them. As a student in computer science, my focus had already shifted to industrial engineering. PricewaterhouseCoopers added a focus on Accountancy to this. Looking back, I can say that the biggest challenge of this graduation project was to bring the three domains, computer science, industrial engineering, and accountancy together. With results; this research provides an architectural framework with aspects of all three domains.

I would like to thank a number of people, without whom I would not have managed to bring this project to a satisfying end. First, my gratitude goes to Paul Grefen, my supervisor. More than providing me with the necessary expertise at the right time, and challenging me to find solutions on my own, he showed me of what I am capable and what I have learned during my graduation project. Second, I would like to thank Wijnand Braal, my supervisor at PricewaterhouseCoopers, in particular for his patience. Endless discussions we had and questions I asked, to get a grip on the accountancy domain and the processes at health insurers.

Furthermore, my gratitude goes to Rens Bastiaansen, who also supervised me at PricewaterhouseCoopers. He has put a lot of effort in explaining the use of IT within the accountancy domain and provided me with new insights when I lost overview. To guarantee the contribution of computer science in this project, Natalia Sidorova was involved. I would like to thank Natalia for her contributions. My gratitude also goes to Ad Aerts, who stood in for Natalia in the last stage of this project.

At a Dutch health insurer, who shall be nameless, I had the opportunity to carry out a case study to gain insights in its processes, for which I am really grateful. By providing me with a workstation to take a look in their systems and applications, and giving me the possibilities for interviewing employees, I gained experience with modeling business processes ‘in real life’.

I also would like to thank my parents and sisters, who were ever supportive during my studies, regardless of what sideline activities I did to postpone my graduation. I also had lots of support from my friends, specially in the last, stressful weeks. Special thanks go out to Bram Kater, for his many useful comments and all the help he provided during this project.

Contents

Chapter 1

Introduction

Today, accountants find a real challenge in performing audits. During an audit, an entity's accounts and financial situation is examined to verify the accurateness and fair presentation of the financial situation by the financial statements. Scandals like Enron, Worldcom, and Ahold and the huge penalties imposed on fraud, illustrate the importance of the audit.

An audit consists of two phases: the interim audit and the final audit. During the interim audit, systems and processes are investigated to address the level of internal control. Internal control is a process, effected by an entity's management and employees, designed to give reasonable assurance on achieving goals concerning effectiveness and efficiency of operations, reliability of financial reporting, and compliance with laws and regulations [COS04]. During the final audit, the financial statements are checked and analyzed. Dependent on the level of internal control, the financial statements are analyzed more or less extensively; a low level of internal control implies more financial data analysis due to increased risk on financial misstatements.

Information technology provides new opportunities in the field of auditing. Although audit-supported software is generally accepted, the audit is performed manually to a large extent. Audit-supported software is only deployed after having manually collected data and is basically based on data analysis. A situation where the audit is performed automatically, is desirable for several reasons. Information systems are getting more complex these days and it becomes difficult to audit these systems. Problems are foreseen in the future with less people being educated as accountant. And more timely (audit) information is needed to cope with rapidly changing and competitive global markets.

The possibilities of IT-enabled auditing are subject of discussion for a while. Nevertheless, big improvements on using IT with audits are not being achieved due to the complexity of the problem and the stiffness of the audit profession. Questions are raised on the feasibility of IT-enabled auditing, on how IT-enabled auditing should look, on which technology to use for IT-enabled auditing, and on how to assign responsibilities with IT-enabled auditing. Could IT-enabled auditing lead to continuous auditing instead of performing the audit once a year, is also questioned.

The objective of this graduation project is to investigate the possibilities for IT-enabled and, as a consequence, continuous auditing. Due to the complexity of auditing, each industry requires a different audit approach, therefore we have chosen to focus on IT-enabled and continuous auditing at (Dutch) health insurers. Processes at health insurers are already highly automated, a restriction for applying IT-enabled auditing (automatically audit manual procedures and handwritten forms does not make sense). By designing a system capable of the IT-enabled and continuous auditing of health insurers, we try to answer the questions stated above.

1.1 PricewaterhouseCoopers

This research has been conducted at PricewaterhouseCoopers (PwC). PwC's main activities are developing and offering services and solutions on business and industry issues, mainly in the field of assurance, tax, human resources, transactions, performance improvement, and crisis management. PwC is active in all industries and delivers services to smaller companies in the private sector and organizations in the public and non-profit sector.

PwC is divided into four service domains: Assurance, Tax & Human Resource Services (HRS), Advisory and Firm Services, which offers internal services to PwC. The four domains are referred to as Lines of Service (LoS). PwC clearly separates its audit activities, also called channel 1 activities, from its advice activities, channel 2 activities, where only one of the two activities can be conducted for a client at the same time. The internal structure of PwC can be described by the matrix given in Figure 1.1. Besides the Lines of Services, the industries PwC operates in are also displayed.

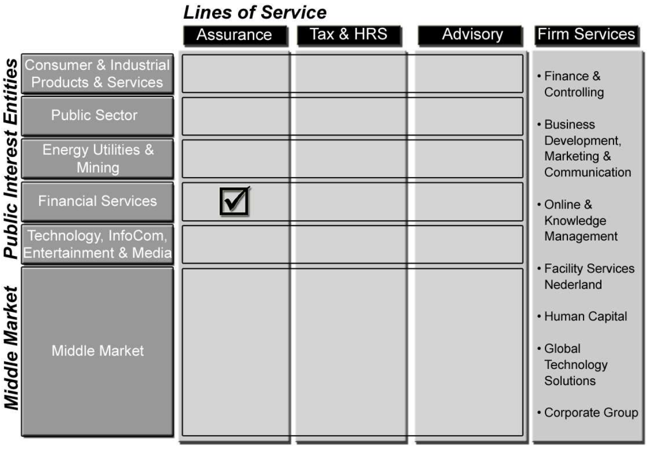
Lines of Service
Assurance
Tax & HRS
Advisory
Firm Services
Public Interest Entities
Consumer & Industrial Products & Services
Public Sector
Energy Utilities & Mining
Financial Services
Technology, InfoCom, Entertainment & Media
Middle Market
Middle Market
☑
• Finance & Controlling
• Business Development, Marketing & Communication
• Online & Knowledge Management
• Facility Services Nederland
• Human Capital
• Global Technology Solutions
• Corporate Group

Figure 1.1: The internal structure of PwC

During my graduation project, I have been positioned at the Systems and Process Assurance (SPA) department, which is a subdivision of the LoS Assurance. This LoS engages mainly accountants, its main task being to perform financial audits for clients. SPA is specialized in auditing complex, automated systems and assists audit and non-audit clients in the field of risk control and internal control improvement. Regarding to the industry, this graduation project is performed at Financial Services, covering banks and insurers.

Chapter 2

Background

The previous chapter introduced briefly the questions that have been investigated in this research. A short introduction into audits and their dependence on internal control, has been given. This chapter provides a more extensive description of the audit process, Section 2.1 and internal control, Section 2.2. Continuous assurance is currently a hot topic within the accountancy domain. Continuous assurance leans on continuous auditing, which introduces an additional reason to investigate the possibilities for continuous auditing. Continuous assurance is explained in Section 2.3. Section 2.4 discusses the current state of the art of IT-enabled and continuous auditing.

2.1 The audit process

An accountant's main activity is performing audits. Audits can have various subjects, most known subjects are financial audits and compliance audits. Other subjects include IT audits and quality audits, but this thesis focuses on financial and compliance audits. The goal of a financial audit is to provide an independent opinion on the relevance, accurateness and completeness of the financial statements of an organization and if the financial statements are fairly presented. A compliance audit is intended to provide an independent opinion on the effectiveness and correctness of the actions taken to comply with law and regulations.

Different phases are distinguished within an audit process [Pri04]: analyze risks, build team, establish plan, build evidence, and complete. Build evidence, where the actual audit takes place, consist of two parts: the interim audit and the final audit. Figure 2.1 provides an overview of the phases, each phase is described below.

Analyze risks Before an audit agreement is established, risks are analyzed by taking standardized interviews. For example management integrity and going concern continuation are taken into account, to decide whether or not the accountant is willing to perform the audit.

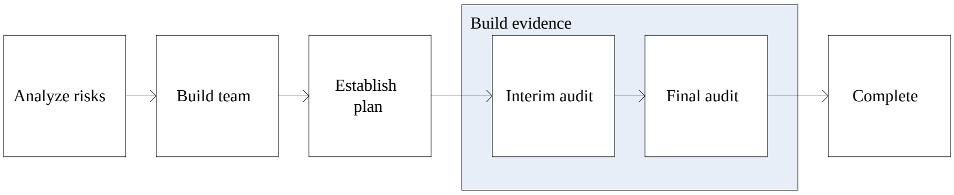

Figure 2.1: Audit process

Build team A team that will perform the audit is composed, with experts in relevant fields, e.g. IT-auditors if IT systems are complex. Different risks found during the analysis, lead to different team compositions.

Establish plan An audit plan is formulated in accordance with the client. The plan contains among other things an audit approach, a client communication plan, budget information, a time schedule, and deliverables.

Build evidence If the agreement and the audit plan are established, evidence is collected. Evidence is collected from the organization's documentation, interviews, and "show me"-meetings in which the accountant asks employees to show their work processes. In this phase the actual audit is executed, consisting of two parts:

Interim audit IT systems and processes are checked during this audit, to reveal the level of internal control, see Section 2.2, and the risks related to the systems and processes. Financial data related to systems and processes with higher risks, are more extensively investigated during the final audit.

Final audit The final audit focuses on (financial) data analysis by random checks and statistical data analysis. Furthermore, the financial statements are checked to be in conformity with national standards, e.g. General Accepted Accounting Principles (GAAP) or International Financial Reporting Standards (IFRS).

Complete During the completion, all findings are reported, and an opinion is formed and communicated in the form of an auditors' certificate.

An accountant provides an opinion based on the assessment of internal control measures of an organization. The internal control measures are investigated at random, therefore an audit does not provide a completely reliable opinion. With the current audit approach, a completely reliable opinion is not feasible.

An audit provides an independent opinion by a third party and is mostly performed once a year. An organization needs information on the relevance, accurateness, completeness, and fair presentation of its financial statements also during the year. An internal audit, performed by the organization itself, is executed to this end. To the audit performed by a third party is often referred as the external audit.

2.2 Internal control

The previous section explained that an external audit is found on an organization's internal control. Internal control is broadly defined by the Committee of Sponsoring Organizations of the Treadway Commission (COSO) [COS04]:

> "A process, effected by an entity's board of directors, management and other personnel, designed to provide reasonable assurance regarding the achievement of objectives in the following categories:
>
> - Effectiveness and efficiency of operations.
> - Reliability of financial reporting.
> - Compliance with applicable laws and regulations."

In 2004, COSO developed the COSO Enterprise Risk Management framework, to provide a structured approach for internal control activities. The framework is an extension of the COSO internal control framework, developed in 1992. To the categories of objectives stated above, a category is added, namely "Conformity to the organization's strategy". Figure 2.2 provides an overview of the framework, it relates the objective categories to the business risks and the internal control system, and distinguishes four organization levels for which internal control activities are designed. The following components are distinguished [COS04]:

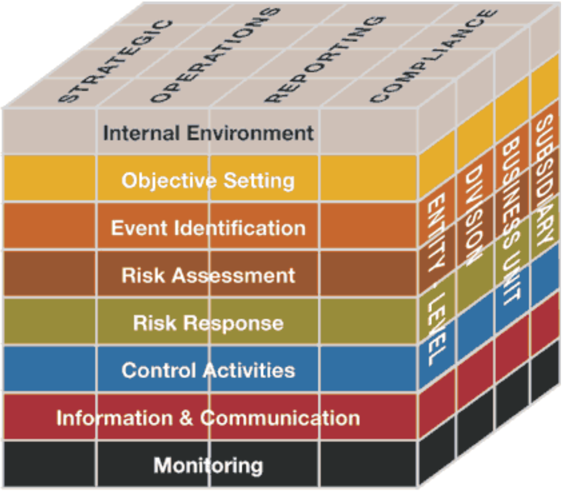

Figure 2.2: COSO Enterprise Risk Management framework [COS04]

Internal environment The organization's environment is the foundation for all other components of enterprise risk management, providing discipline and structure. The internal environment comprises e.g. ethical values, managements operating style and how it assigns authority and responsibility.

Objective setting For an organization's mission or vision, management establishes strategic objectives, selects strategy, and establishes related objectives at different levels of the enterprise, aligned with and linked to the strategy.

Event identification Management recognizes that uncertainties exist: it cannot know with certainty whether and when an event will occur, or its outcome should it occur. As part of event identification, management considers external and internal factors, e.g. economic environment, technological factors, and personnel, that affect event occurrence.

Risk assessment During risk assessments, potential events are analyzed to investigate their influence on the achievement of objectives. Management assesses events from two perspectives: likelihood, the possibility that a given event will occur, and impact, the effect of an event, should it occur.

Risk response Possible risk responses are identified and their effect on event likelihood and impact, in relation to risk tolerances and costs versus benefits, are considered.

Control activities Control activities are the policies and procedures for executing risk responses properly. Control activities occur at all levels in an organization, and are part of the process by which an organization strives to achieve its business objectives. Relying on complex information systems these days, introduces a necessity for information systems controls. Two groups of controls are distinguished: application controls, built within applications, and general IT controls, which are controls over information technology management, e.g. security management and software acquisition. These controls are combined with manual process controls where necessary.

Information and communication External and internal information is identified, captured and communicated in a form and timeframe that enable personnel to carry out their responsibilities. Effective communication also occurs in a broader sense, throughout the organization and to external parties. Information is needed at all levels of an organization to identify, assess and respond to risks.

Monitoring Enterprise risk management is monitored, to address the functioning of its components and the quality of their performance over time.

During an audit, to investigate the level of internal control of an organization, internal controls are checked on their design, existence and operating effectiveness.

2.3 Continuous assurance

This section explains briefly the relation between continuous assurance and continuous auditing.

In the previous section, internal controls and underlying risks were discussed. An increasing need for timely and ongoing information on the effectiveness of risk management and control systems, is observed. In other words, organizations have a need for continuous assurance.

Continuous assurance addresses the need for assuring that the controls are working properly, that significant risk are identified, and that significant violations of regulations or irregularities have not occurred [Cod05].

Continuous assurance can be achieved through continuous auditing and continuous monitoring. Continuous monitoring are processes put in place by management to ensure that the policies, procedures and business processes are operating effectively. Management identifies critical control points and implements automated tests to determine the proper working of controls. Continuous monitoring involves the automated testing of all transactions and system activities, within a given process area, against control rules [Cod05]. Continuous auditing is a methodology that enables independent accountants to provide written assurance on a subject matter using a series of auditors' reports issued simultaneously with, or a short period of time after, the occurrence of events underlying the subject matter [CIC99]. There is an inverse relationship between the adequacy of management's monitoring and risk management activities and the extent to which accountants perform detailed testing of controls and assessment of risks [Cod05].

Advantages of continuous assurance include [Cod05]:

- Increased confidence in financial results;
- Improvement of financial reporting operations;
- Reduced financial error and potential for fraud;
- Reduction of revenue leakage;
- Sustainable and cost-effective means to support compliance.

2.4 State of the art

This section discusses the current state of the art of continuous auditing. A uniform view on continuous auditing is not available, research into the following technologies is currently conducted:

- Embedded audit modules;
- Business/artificial intelligence;
- Statistical modeling;
- Trend analysis;
- Primary key indicators;
- Monitoring high-risk transactions.

Short descriptions of agent-based continuous auditing, of continuous auditing using web services, and data mining in the continuous auditing process are given. Also XBRL is briefly discussed, to explain its contribution to continuous auditing.

2.4.1 Agents-based continuous auditing

A digital agent is a composition of software and data, moving from one computer or network to another, continuing its execution. In a continuous audit environment, an agent performs services related to the subject matter being audited, on behalf of the accountant [WS01]. Agents can do testing, normally done by accountants, off-site. Audit routines can be designed and executed remotely by agents, to test transactions and controls continuously. An agent can be reactive or proactive. Reactive agents stay in a defined location on a system, use predetermined procedures, and trigger alerts to the accountant when irregularities occur in the system. Proactive audit agents are free to move through systems and networks and operate with intelligence. They are able to look in a client's database, determine appropriate audit routines and acceptable level of errors, run audit routines, determine control status and generate reports [GF99].

2.4.2 Continuous auditing using web services

The article "A continuous auditing web services model for xml-based accounting systems" discusses the use of XML and web services to provide a continuous auditing web service (CAWS) [MG04]. The CAWS mechanism runs as a web service at the accountancy firm's computing environment and can be invoked by an organization to provide assurance on specific business processes, on financial reporting, or on the operation of internal controls. In this article, an architecture for accomplishing a pull model of continuous auditing is proposed. A requirement of this system is, that the client needs to specify its processes in BPEL4WS, which is a language for the formal specification of business processes and interaction protocols. How data is checked within the CAWS system, is only briefly mentioned. For data checking, the use of existing tools in a web service environment, is emphasized.

2.4.3 Continuous auditing using audit data marts

Audit data marts are small data warehouses containing audit-related information, extracted for further inspection [RSE02]. Required data is obtained from the data mart, it is filled periodically and automatically by downloading data from the client's database. Therefore, the auditing is claimed to be continuous. However, audit functionality used with audit data marts, are mainly based on statistical data analysis for recognizing patterns and trends in data. Historical data is needed to perform analyses, which makes real continuous auditing less plausible.

2.4.4 XBRL

eXtensible business Reporting Language (XBRL) is an open standard for composing electronic reports and data exchange by means of the internet, based on XML [XBR06]. XBRL is a standard for storage and exchange of financial data for external financial reporting, for internal reporting and management information provisioning and reporting to external stakeholders, e.g. tax authorities. XBRL separates content from representation and uses “tags” to identify the meaning of data. XBRL uses taxonomies in which reporting elements are defined and the relation between elements within a taxonomy or in other taxonomies. Taxonomies are divided in generic taxonomies, providing international reporting standards, and company specific taxonomies. Different generic taxonomies are available, which makes it possible to handle data in different languages and accounting standards. Since 2007, Dutch companies can file financial reports in the XBRL format at the Chamber of Commerce.

The next chapter discusses the goal and approach of this research.

Chapter 3

Research goal and approach

As mentioned in the introduction, IT offers new possibilities for the accountancy domain, more possibilities than currently applied. There appears to be a lack of practical knowledge on how to apply current IT technologies in the accountancy domain. Chapter 2 provided information on the audit profession and gave a brief overview of current theoretical research in the field of IT-enabled and continuous auditing. Our research focuses on a more practical insight of IT-enabled and continuous auditing; an architectural framework for IT-enabled, continuous auditing at health insurers is provided. In this chapter, the research objectives are formulated (Section 3.1), the design approach is mentioned (Section 3.2), the research scope is indicated (Section 3.3), the research questions and deliverables are framed (Section 3.4), a research approach is provided(Section 3.5), and a report outline is given (Section 3.6.

3.1 Problems and objectives

The aim of this research is to gain insights into the possibilities of IT-enabled, continuous auditing, particularly of health insurers. This is achieved by defining an architectural framework for IT-enabled, continuous auditing. A framework structures architecture description techniques by identifying and relating various architectural viewpoints and their associated modeling techniques [LTP^{+}05]. A description of architectures and architectural viewpoints, is given in Subsection 3.1.2.

The starting point for the architectural framework for IT-enabled, continuous auditing, is a point in time ten years from now. As a result, we are not interfered by current restrictions caused by available technology. Before designing the architectural framework, the present situation is analyzed to gain insights into the processes of health insurers, of importance for the audit performed at health insurers. For this goal, a case study is performed at a health insurer in the Netherlands.

PwC as major accountancy firm, wants to be ahead of its competitors and desires to be a trusted advisor for its customers. To this end, PwC needs to keep up with current developments in its domain. IT-enabled auditing and continuous auditing are hot topics at the moment, which gives additional cause for this research. Other advantages for PwC and the accountancy domain in general are already discussed in Section 2.3.

3.1.1 Main question and deliverables

The main question for this research is:

> In what way can PwC, by means of IT, automate its audit process and provide new services for the health insurer industry, with a view to provide continuous assurance as defined by long-term strategy?

The following main deliverables are identified in this research, viewpoints are discussed in Subsection 3.1.2:

1. A functional description of the processes of health insurers, relevant to the audit.
2. A requirements analysis for the audit system to be designed for IT-enabled, continuous auditing at health insurers.
3. A functional description of the audit system to be designed for IT-enabled, continuous auditing at health insurers. This includes a process viewpoint of the audit system.
4. A data and organization viewpoint of the audit system to be designed for IT-enabled, continuous auditing at health insurers.
5. A system viewpoint of the audit system to be designed for IT-enabled, continuous auditing at health insurers.
6. An opinion on the feasibility of IT-enabled, continuous auditing and a proposal for managing changes towards an audit system for IT-enabled, continuous auditing.

3.1.2 Enterprise architecture as operationalization

The complexity of today's organizations demands for the alignment of business and IT. A structured approach is desirable and using a frame of reference makes communication on business and IT alignment more efficient. To create an organized overview of the organization's structure, its business processes and the technical infrastructure, a way to express the different aspects and domains, and their relations has to be available [LTP$^+$05]. This subsection is concerned with explaining the term enterprise architecture and its application, intended for the goals mentioned above.

Architecture is defined by IEEE Standard 1471-2000 as [IEE00]:

> “The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principle guiding its design and evolution”

Currently, architectures are often used at the level of a complete organization instead of only in the IT domain. This leads to a definition of enterprise architecture [LTP$^+$05]:

> “Enterprise architecture is a coherent whole of principles, methods and models that are used in the design and realization of an enterprise’s organizational structure, business processes, information systems and infrastructure.”

Enterprise architecture captures the essentials of business and IT, rather than specific, currently available solutions. It creates an integrated perspective of an enterprise, with techniques for describing architectures in a coherent way and viewpoints for communicating architectures to different stakeholders. Moreover, enterprise architecture is used for positioning new developments within the context of existing processes, systems and other assets of an organization and it serves the purpose of complying to laws and regulations, which increasingly demand organizations to have a clear insight into their operations [LTP$^+$05].

Enterprise architecture makes an effort to divide visualization from content: the content is modeled, i.e. an abstracted and unambiguous conception of a domain is given, and for each stakeholder a set of models are provided that result in a view [LTP$^+$05]. A view is defined by the IEEE 1471 standard as “a representation of a system from the perspective of a related set of concerns” [IEE00]. A viewpoint is defined as “a specification of the conventions for constructing and using views” [IEE00].

3.2 Architectural framework as methodology

In the context of architecture descriptions, many viewpoint frameworks are available. Truijens distinguishes the following viewpoints [TOM$^+$90]:

Data Describes the data structure, the meaning of the data, and the storage of the data within the architecture.

Systems Describes the system components of an IT system.

Organization Describes the responsibility of parties within an organization for different aspects of the architecture. The organization viewpoint affects all the other viewpoints.

Communication Describes the communication between IT systems.

Configuration Describes the configuration of the hardware.

In the article "Architectural Blueprints" by Kruchten, a different viewpoint framework is described, the *'4+1' view model* [Kru95]. The viewpoints are similar to those of Truijens's framework, accept that an additional viewpoint for processes is described. The process viewpoint indicates the activities of processes within the IT system and the relation of the activities with the other viewpoints. The process viewpoint is seen as the central part of the architecture and serves as a starting point for the other viewpoints.

For this research, we also take the process viewpoint as a starting point. The goal of this research is to gain insights into the possibilities of IT-enabled, continuous auditing, specifically at health insurers, by defining an architectural framework and not to provide a full implementation for an IT-enabled, continuous auditing system, which implies that hardware configuration is not relevant. For that reason, the configuration viewpoint is left out. The communication between IT systems is also of more importance for an implementation, therefore the communication is only briefly mentioned as part of the systems viewpoint. To summarize, for the architectural framework subject of this thesis, we design a *process viewpoint*, a *data viewpoint*, an *organization viewpoint*, and a *system viewpoint*.

For modeling enterprise architectures, a score of languages, methodologies and techniques exist, different for each domain. In the light of this research, bringing the domain of accountancy, industrial engineering and IT together, generally accepted methodologies and techniques with widely available background information are used. These are explained when needed.

3.3 Research scope

In the previous chapter, the PwC's audit process is discussed. Two phases within the audit process are distinguished: the interim audit and the final audit. The audit depends on the internal control of the entity, subject to the audit. Internal control focuses on providing assurance on the achievement of objectives concerning efficiency and effectiveness of operations, accuracy of financial reporting and compliance to current laws and regulations. Within internal control, two kinds of control activities are distinguished: application controls and manual controls. Application controls are build into computer applications (automated controls), whereas manual controls depend upon one or more individuals for their application.

For application controls to function correctly, they depend on general IT controls. General IT controls are the broad entity-level information processing controls, covering hardware access as well as system and application software development, change, and maintenance. For manual controls to function properly, the control environment must be designed for supporting manual controls [CCA06]. The control environment establishes the tone of an organization, influencing the control consciousness of people. Control environment factors include the integrity, ethical values and competence of the organization's people, the way management assigns authority and responsibility (separation of duties), and management's philosophy and operating style [COS04].

Figure 3.1 gives an overview of the relation of the audit, internal control and types of control activities. It also distinguishes the three categories, efficiency and effectiveness of operations, reliability of financial reporting, and compliance with applicable laws and regulations, which can be subjects of an audit.

This research focuses mainly on audits on financial reporting and compliance. We take efficiency and effectiveness concisely into consideration. As stated in Section 2.2, the audit depends on internal control. For IT-enabled, continuous auditing, focus is on application controls, since information on application controls is digitally available. How to manage non-automated internal control within IT-enabled, continuous auditing, is superficially mentioned.

For the internal control, the emphasis is on application controls since they can be managed automatically. How to manage manual controls, general IT controls, and the control environment within the audit system for IT-enabled, continuous auditing, is superficially handled.

As shown in Figure 3.1, three levels are distinguished: strategic, tactical, and operational. They refer to levels in an organization. At the strategic level, the overall strategy of an organization and its policies are decided. These decisions are communicated to the tactical level, where they are translated into input that can be directly used at the operational level. Details that are not captured at the strategic level, are filled in at tactical level. At the operational level, the operational processes of an organization are performed. Figure 3.1 shows that within an organization the audit is performed at strategic level, since it is a process only executed once a year. Parts of an audit can be repeated more often, which makes them tactical processes. Internal control is a tactical process, which defines the operational processes implemented by application controls and manual controls. This can be seen as the interpretation of internal control at the operational level. The general IT controls and the control environment are both process independent and therefore not classified into a level.

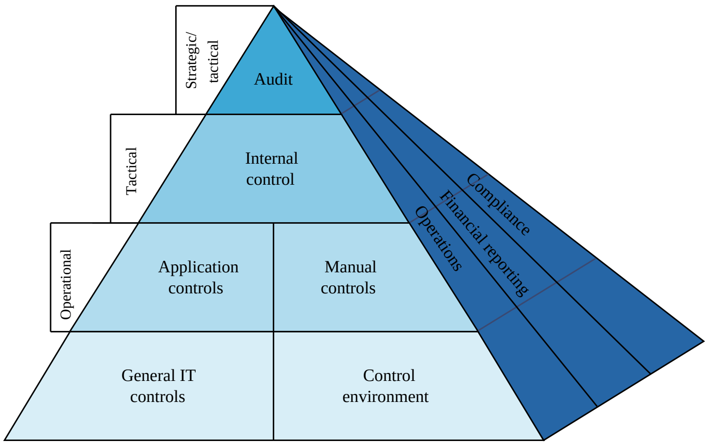

Figure 3.1: Framework of audit dependencies and scope

As already mentioned, subject of this research is the development of an architectural framework for continuous, IT-enabled auditing. The focus for developing this framework, is a situation in the future, ten years from now, to be independent of current technical limitations. Concerning the audit, we mainly focus on the interim audit. To check if the financial report is framed conforming the international financial reporting standards, is left out.

3.4 Research questions and deliverables

The following questions are used to structure the results of this research, in order to accomplish the research objective. The questions are related to the main deliverables, the first number corresponds to the deliverable mentioned in Subsection 3.1.1.

1.1 How are the processes, of concern for the PwC's audit process, at health insurers organized?
2.1 What are the requirements of the health insurer for applying IT-enabled, continuous auditing?
2.2 What are PwC's requirements for an IT-enabled, continuous auditing system (from an accountant's viewpoint)?
2.3 What are the technical requirements for an IT-enabled, continuous audit system?
3.1 What functionality does the IT-enabled, continuous audit system provide?
3.2 How does the IT-enabled, continuous audit system relate to processes at the health insurer?
3.3 What processes are distinguished within the IT-enabled, continuous audit system?
4.1 How is the data structured, that is used and produced by the IT-enabled, continuous audit system?
4.2 How are responsibilities of the stakeholders of the IT-enabled, continuous audit system assigned?
5.1 What system components are used for an implementation of the IT-enabled, continuous audit system and how are they structured?
5.2 What technology can be used for the system components?
5.3 How is the communication between the health insurer and the IT-enabled, continuous audit system achieved?
6.1 How does a change management plan for an implementation of the IT-enabled, continuous audit system look?
6.2 Is an IT-enabled, continuous audit system feasible?

The corresponding deliverables, extensions of the main deliverables in Subsection 3.1.1, are:

1.1 A functional description in a formal modeling language.
2.1 Use cases and user requirements.
2.2 Use cases and user requirements.
2.3 System requirements for each architectural viewpoint.
3.1 A functional description in a formal modeling language.
3.2 A functional description in a formal modeling language.
3.3 Process descriptions in a formal modeling language.
4.1 Data descriptions in a formal modeling language.
4.2 Overview of assignments of responsibilities to activities defined in the functional description.
5.1 A system architecture using architecture patterns.
5.2 Recommended technologies for each component in the software architecture.
5.3 Recommended communication technologies.

6.1 A structured change management plan.
6.2 An opinion formed through the execution of this research.

3.5 Research approach

This research is conducted in the phases listed below. The research approach is based on the recommendations of Kempen and Keizer [KK00] and Van Aken et al. [ABB01] for an industrial engineering research project. The deliverables in Section 3.4 are related to the phase in which they are delivered.

1. Project planning and refinement;
2. Literature research;
3. Analysis present situation (1.1);
4. Design requirements future situation (2.1, 2.2, 2.3);
5. Functional description design future situation (3.1, 3.2);
6. Architectural framework design future situation (3.3, 4.1, 4.2, 5.1, 5.2, 5.3);
7. Change management plan future situation (6.1);
8. Conclusions and recommendations (6.2);
9. Reporting and presentation.

The activities of the phases mentioned above, are not executed sequentially. Figure 3.2 provides an overview of the execution sequence of the activities. First, during “Analysis present situation”, the processes of the health insurer, related to the audit, are mapped out. To this means, a case study at a health insurer is performed. When insights in the processes at health insurers are gained, requirements for the IT-enabled, continuous audit are elicited. These requirements can only be gathered after having a view on the health insurers’ processes of importance for audits currently performed at health insurers.

Requirements are iteratively gathered, first the requirements for the functional description and process viewpoint of the audit system for IT-enabled, continuous auditing are elicited and analyzed. Since the process viewpoint is the central part of the architectural framework to be designed, the requirements of the other viewpoints are based on the design of the process viewpoint. During the design of each viewpoint, decisions could be made that affect the viewpoints still to be developed. Therefore, after each viewpoint design, the requirements of the next viewpoint to be designed are analyzed and adapted if necessary.

When the architectural framework is designed, a plan for implementing the system in the future is provided. Also questions on the achievability of IT-enabled, continuous auditing, are answered.

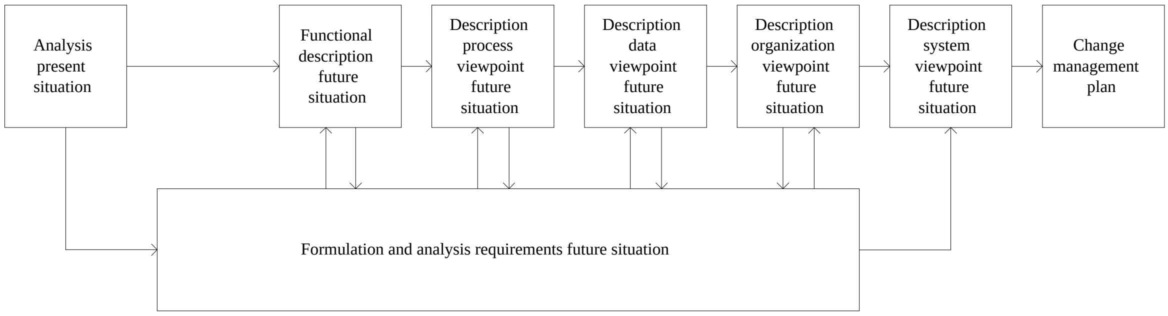
Analysis present situation
Functional description future situation
Description process viewpoint future situation
Description data viewpoint future situation
Description organization viewpoint future situation
Description system viewpoint future situation
Change management plan
Formulation and analysis requirements future situation

Figure 3.2: Research phasing

3.6 Report outline

In this chapter, the research objectives, scope and approach, and design approach are discussed. Chapter 4 provides the analysis of the processes at health insurers, supported by a case study at a Dutch health insurer. Chapter 5 gives use cases, user requirements and system requirements. In Chapter 6 the functional description and the process viewpoint of the IT-enabled, continuous audit system are given. Also the connection with the processes at health insurers is discussed. Chapter 7 discusses the data viewpoint and organization viewpoint. Organizational issues foreseen with the audit system for IT-enabled, continuous auditing are also discussed in this chapter. Chapter 8 gives the system viewpoint by means of a system architecture. It also mentions the communication between the health insurer and the audit system for IT-enabled, continuous auditing. Chapter 9 recommends an implementation plan and concludes with answering the question “Is IT-enabled, continuous auditing feasible?”.

Chapter 4

Present situation — Functional description

In this chapter, a functional description is provided of processes relevant for the audit of the health insurer, see Section 4.1. The functional description is useful to gain insights into the processes of the health insurer to base the audit system for IT-enabled, continuous auditing upon. The functional description is based on a case study at a health insurer in the Netherlands and internal information of PwC on health insurers. When analyzing processes at the health insurer, we gradually realized that mapping out all the processes within a health insurer would be too complex, time consuming, and irrelevant for the aim of this research, therefore we decided to focus solely on the premium process as subject for the audit system for IT-enabled, continuous auditing. A detailed overview of the premium process is given in Section 4.2.

4.1 Health insurer

In the Netherlands, the system for health insurance consists of three compartments: the AWBZ (general law special medical expenses), the standard insurance, and the additional insurance. The AWBZ is an insurance for all citizens against serious medical risks, e.g. long illness and handicaps. The standard insurance, obligatory for all citizens, includes all necessary health care, e.g. transport by ambulance, consulting a general practitioner, hospitalization. The standard insurance is statutory and insurants acceptance is obligatory. This introduces high risks and costs if a considerable part of the insured people has health problems. Therefore, all insurance companies share the costs resulted from these risks, and a main part is covered by the government through subsidies. For the additional assurance, insurers are free to compile their own additional insurance packages. The packages can be specialized for different groups of people, e.g. the elderly, young people, sportsmen. Also provided are packages for specialized health care, e.g. dental care.

Focus of this research is on the standard and additional insurance, the AWBZ has been left out. Reasons are that implementing the AWBZ differs highly from the execution of the standard and additional insurance and emphasis of present audit is also on the standard and additional insurance.

As already explained in Section 3.3, an organization is divided in three levels: the strategic level, the tactical level, and the operational level. Figure 4.1 gives a three-level overview of the processes within the health insurer, for the execution of the standard and additional insurance. Data Flow Diagrams are used to model the processes and data flow within the health insurer, an overview of this technique is given in Appendix A.

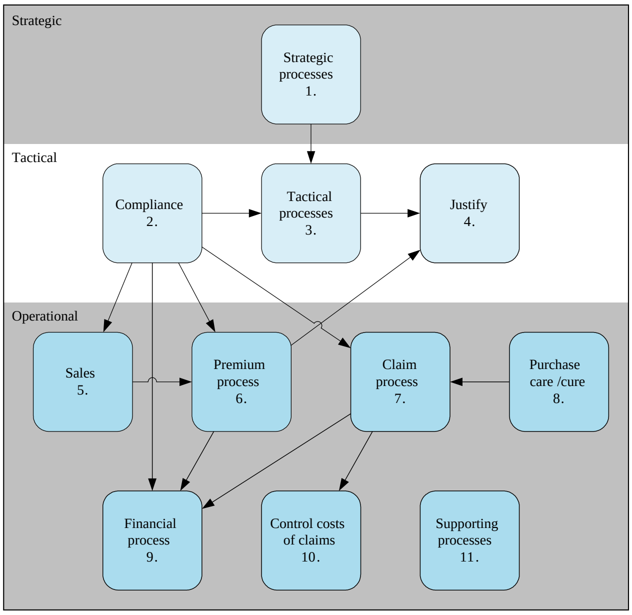

Figure 4.1: Processes health insurer

Strategic processes include deciding the strategy and policies of the health insurer. Translation of strategic decisions to input for operational processes, occurs by tactical processes. Current laws and regulations dictate rules for processes of health insurers, with which they have to comply. Health insurers receive subsidies from the government, based on the characteristics of their insurants, e.g. age, residential area, long illness. The insurer needs to justify that their insurants file is correct and up to date. Tactical processes provide input for all operational processes, to keep the overview simple, the data flows from tactical processes have been left out.

At the operational level, the sales process includes developing new insurances and selling insurances, the premium process contains subscription, mutation and unsubscription of insurants, the claim process handles claims from insurants and care/cure organizations, the purchase care/cure process makes an indication of necessary care and cure and negotiates on prices in advance, and the financial process handles all financial transactions. The control costs of claims process recovers costs from other insurance companies, and performs statistical data analysis on claims to detect fraud from care/cure organizations. Supporting processes support all operational process with e.g. human resources, and IT systems. To give a simple overview of all processes, the data flows from supporting processes have been left out.

Figure 4.2 gives an overview of the tactical and operational processes discussed above, together with the connection with external entities.

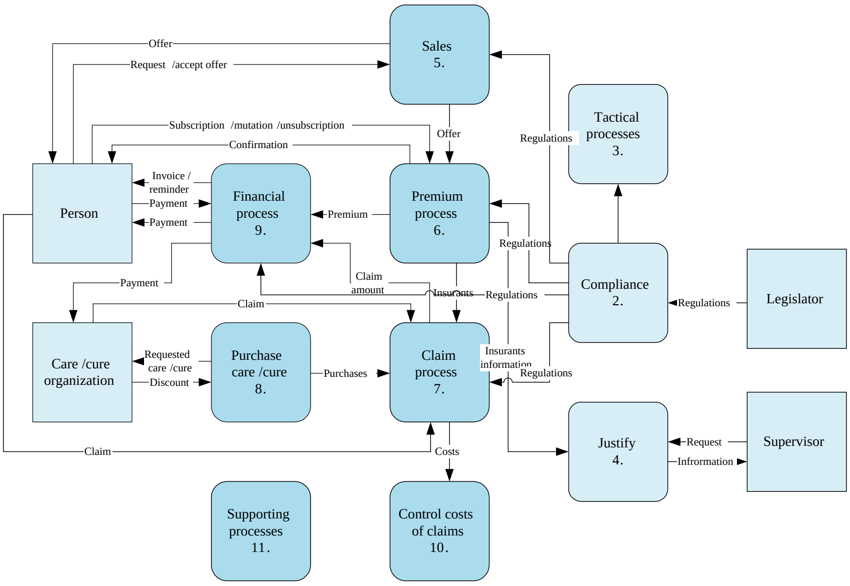

Figure 4.2: Tactical and operational processes health insurer

We have further explored the operational processes and have come to the conclusion that mapping out all processes would be very complex and time consuming. Since the goal of this research is to define an architectural framework for IT-enabled, continuous auditing, and not to extensively examine processes at health insurers, we decided to focus on the premium process for the remainder of this research.

4.2 Premium process

As stated in the previous section, for the remainder of this research, focus is on the premium process of health insurers. Besides the premium process, also the subprocesses of the financial process that are related to the premium process have been taken into account, since the financial process is of great importance for the audit. This section provides a functional description of the premium process and related financial subprocesses.

In Figure 4.3, an overview of the subprocesses within the premium process is given. A distinction is made between the strategic level, the tactical level, and the operational level, similar to the functional description of the health insurer in Section 4.1. The processes at the strategic and tactical level have already been discussed in the previous section, in this overview they specifically relate to the premium process.

At the operational level, the offer process produces offers with prices based on tactical decisions, on requests from individuals or collectivities, and on characteristics of the to be insured

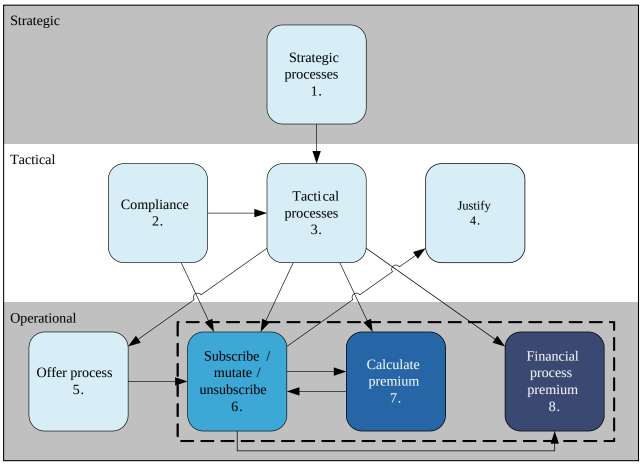

Figure 4.3: Context premium process

person. Based on this offer, a person can accept an insurance and subscribe. Insurants can also ask to mutate their personal information or to unsubscribe. An exact premium calculation is done after subscribing and in some cases after mutating information. The financial process handles collection of the premium and refunding after overpay.

For the audit, the offer process is not of interest. Therefore, focus is on the other three processes. Figure 4.4 gives an overview of the tactical and operational processes discussed above, together with the connection with external entities and data stores that are shared between subprocesses. The GBA is the municipal base administration, containing information on Dutch citizens. When personal information of people is modified, this modification is automatically sent to the health insurer. The premium table contains data for the premium calculation, e.g. the base premium of the standard and additional insurance, discounts for regions, pay frequencies, and collectivities. Authorizations contain access rights to applications implementing the subprocesses. The traditional audit takes only the grey part of the model into consideration, therefore the audit system for IT-enabled, continuous auditing also focuses on this part.

The grey part of Figure 4.4 is further elaborated in Figure 4.5. Sometimes discounts are given on insurances, e.g. when an insurant pays yearly instead of monthly, or when an insurant chooses a higher policy excess. Fines are calculated when the insurant is too late with subscribing. When premiums are prolonged, they are recorded in the general ledger and an invoice to the insurant is sent. When doubtful debtors are processed, outstanding amounts of defaulters are written off as depreciation.

The next chapter provides requirements for an audit software system to support IT-enabled, continuous auditing, applied to the premium process elaborated in this section.

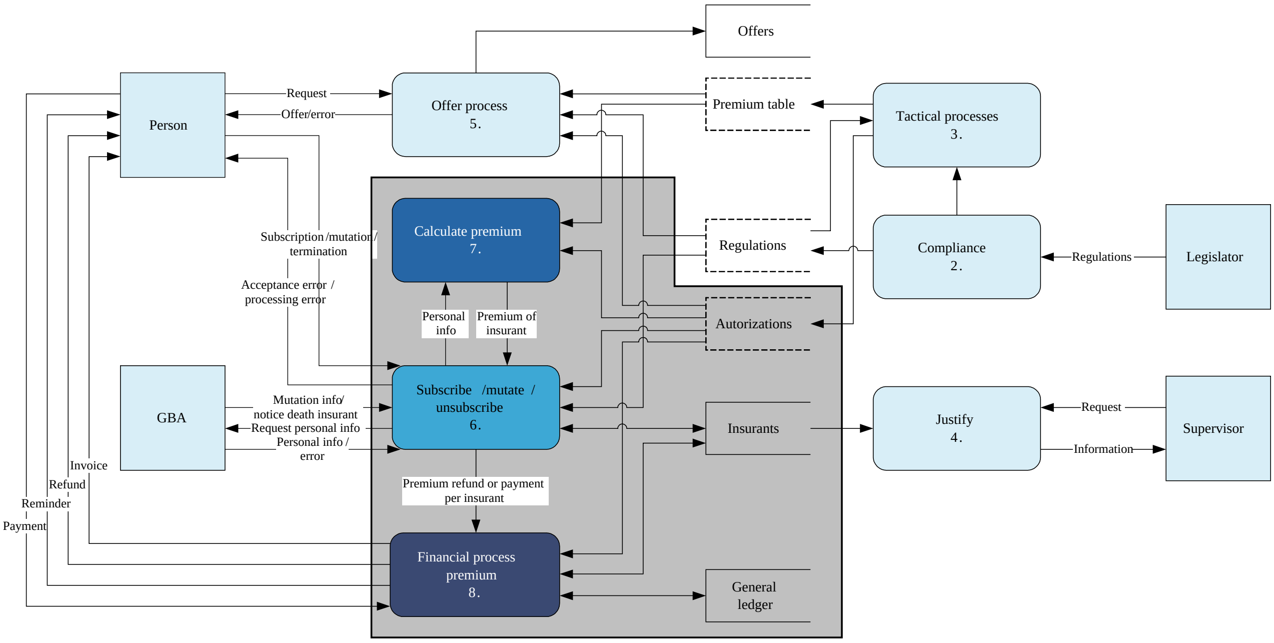

Figure 4.4: Tactical and operational subprocesses of the premium process

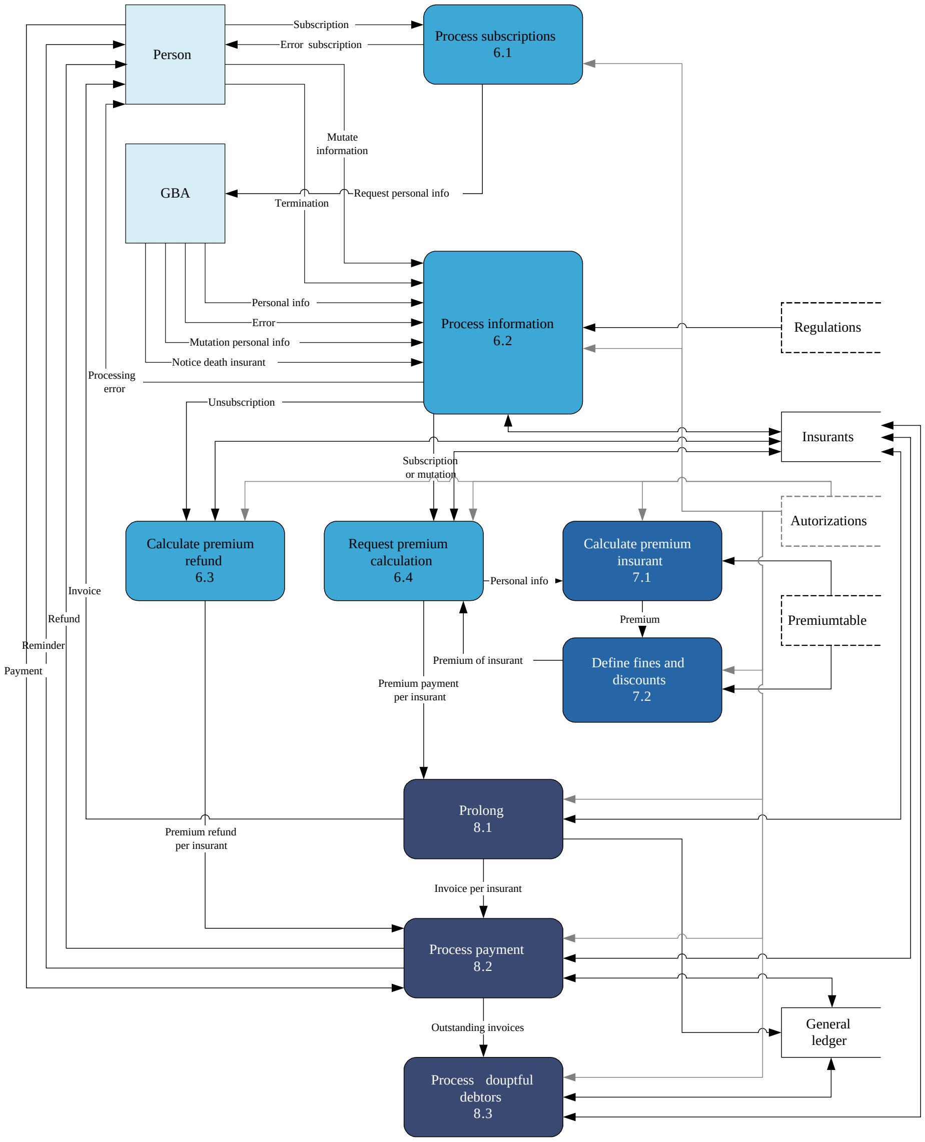

Figure 4.5: Subprocesses operational premium process

Chapter 5

Future situation — Requirements

This chapter is concerned with the requirements of the audit software system to be designed for IT-enabled, continuous auditing, or audit software system in short, mentioned in Chapter 3. As a starting point we take a future point, ten years from now. This gives us the opportunity to design a system, without considering current technical limitations. An introduction into requirements and requirements engineering is given in Section 5.1. Section 5.2 discusses the user requirements, whereas Section 5.3 deals with the system requirements.

5.1 Requirements engineering

The design of a new software system starts with the formulation of the requirements. The success of the system can be derived from the degree to which it meets the intended purpose, with the purpose translated into requirements. Requirements engineering (RE) deals with the process of defining requirements, analyzing and validating the requirements, verifying that the requirements are satisfied by the eventual product and managing change of requirements over time [NE00]. A clear definition is given by Zave [Zav97]:

> "Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families."

Software cannot exist without the system in which it is embedded. Therefore RE is part of systems engineering instead of solely used for software engineering [SJBA98].

RE consists of the following activities, executed iteratively during the system development process [NE00]. The activities are not necessarily executed in the order described below.

1 Eliciting requirements The first step in the RE process, where requirements are collected by analyzing gathered information. The stakeholders of the system and the goals the system should meet, are identified.

2 Modeling and analyzing requirements Fundamental activities of the RE process. The elicitation process uses (parts of) models and their analysis as triggers for further information gathering.

3 Communicating requirements Requirements should be communicated clearly to the different stakeholders in order to analyze, (re-)write and validate them. Therefore, the way requirements are documented plays an important role in RE.

4 Agreeing requirements Requirements should be validated, that is to say it should be determined that the elicited requirements provide an accurate description of the stakeholder's requirements. Difficulties are conflicting goals of different stakeholders and the fact that requirements cannot be proven to be correct through observation, they can only be refuted.

5 Evolving requirements As software systems change in time due to a changing environment, so do the requirements. Therefore managing changing requirements and requirements documentation is a fundamental activity.

Requirements are divided into functional requirements and non-functional requirements. Functional requirements particularly specify the behavior of the system, whereas non-functional requirements specify the overall characteristics, also called the "quality attributes" of the system, e.g. costs, security, reliability, scalability.

For business to adopt the provided solution in the future, strategic alignment is important. To align the business domain, in this case the accountancy, and IT domain, the requirements should be clearly communicated to stakeholders of both domains. Therefore we decided to divide the requirements into user requirements for the accountancy domain, given in Section 5.2, and system requirements for the IT domain, mentioned in Section 5.3. The system requirements specify the internal behavior of the system. For the user requirements, the system is seen as a black box and only the functionality visible to the user is specified.

The next sections contain the requirements of the audit software system, for which this research provides an architectural framework. First the stakeholders and goals are identified following step 1 "Eliciting requirements" and then the requirements are elicited. In Chapter 3, where the architectural framework for this research has been discussed, four viewpoints have been chosen to be highlighted within this framework, which are the process viewpoint, the system viewpoint, the data viewpoint, and the organization viewpoint. Therefore, we divide the functional requirements also according to the four viewpoints.

For the user requirements in Section 5.2, modeling techniques are used to model requirements, as described in step 2 "Modeling and analyzing requirements". This is done to clarify the functionality of the system to the stakeholders. The requirements are written in natural language and communicated to the stakeholders following step 3 "Communicating requirements". Step 4, "Agreeing requirements", and thus validating requirements, is done by two

IT-auditors conversant with auditing health insurers who informally reviewed the requirements. We are aware that requirements that are only informally reviewed by two people with the same background, is too restricted to found a system design upon. However, due to the limited scope of this research, this way of validating is satisfactory for now. Step 5, “Evolving requirements”, is not applicable to this research, due to the short term of this research.

5.2 User requirements

This section is concerned with the requirements of the audit software system for future use from a user viewpoint. The software system itself is seen as a black box and the functionalities of the system, visible to the user, are captured by means of use cases in the form of use case scenarios and corresponding use case diagrams. The user requirements are elicited from interviews with accountants and IT-auditors from PwC, from interviews with employees from a health insurer, and from the book “Bestuurlijke informatieverzorging 2B/toepassingen” [SMJ98].

The stakeholders of the system are the accountant and the health insurer. The accountant provides services to the health insurer, supported by the audit software system. The accountant is responsible for the correct functioning of the system. Cooperation of the health insurer is necessary to obtain the required data.

The goal of the system is to continuously or on a continual basis (more often than once a year), audit a health insurer’s premium process in order to provide an independent opinion on the completeness, accuracy, and validity of items in the accounts related to the premium process and restricted access to financial systems used by the premium process. In addition, an independent opinion on the compliance of the premium process with law and regulations has to be provided.

5.2.1 Use case scenarios and diagrams

To clarify the required functionalities of the audit software system, visible to the user, we provide use case scenarios of the requirements from the stakeholders’ viewpoint. Use case scenarios are short stories with examples of the system functionality. These examples do not give a complete overview of the required functionality, they only provide a feeling of what the system is supposed to do. For each use case scenario, a UML use case diagram has been provided. Use case diagrams are graphical overviews describing the interaction between the system and the user. They are a means to explain the system functionality to the stakeholders. UML use case diagrams are further explained in Appendix B.

Request assurance

The health insurer needs assurance on its financial situation of the past financial year and requests an opinion on the completeness, accuracy, and validity of items in the accounts related to the premium process and restricted access to financial systems used by the premium process, for the past year. The health insurer does not want the accountant to examine more than necessary, therefore the accountant investigates the level of internal control, discussed in Section 2.2, at the health insurer and informs the audit software system on how thoroughly the health insurer's premium process should be checked. The level of internal control affects the audit software system only internally; the lower the level of internal control, the more extensive the audit will be. This does not affect the functionality visible to the users, though it is mentioned here since it is required by the health insurer.

When the system has checked the premium process, the findings are reported to the accountant who provides the health insurer with a report in a useful and readable format. The findings can also be reported directly to the health insurer, in the output format of the system. The findings can range from only the opinion to a detailed overview of checks per process.

A use case diagram for this scenario is provided in Figure 5.1.

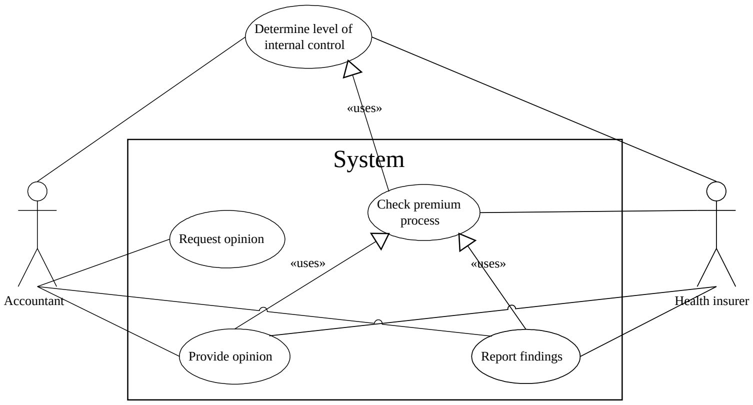

Figure 5.1: Use case "Request assurance"

Request an opinion on compliance

The health insurer requests an independent opinion on the compliance of the premium process with law and regulations for the past financial year. The fact that the health insurer does not want the accountant to examine more than necessary, holds also for this scenario. Therefore the accountant investigates the level of internal control at the health insurer and informs the audit software system on how thoroughly the health insurer's premium process should be checked. The accountant also tells the system which current laws and regulations apply. When the system has checked the premium process, the findings are reported similar to the use case scenario on "Request assurance".

A use case diagram for this scenario is provided in Figure 5.2.

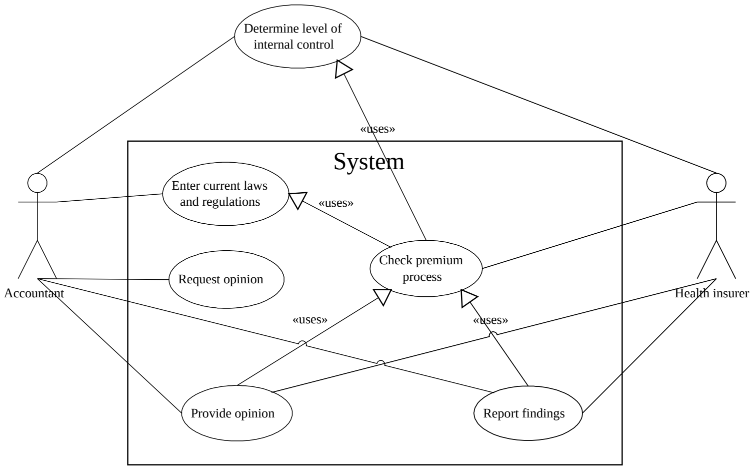

Figure 5.2: Use case "Request an opinion on compliance"

Remediation

The findings of the system may reveal several problems with the internal processes of the health insurer. Some subprocesses of the premium process could not function correctly. The accountant communicates this to the health insurer, and checks after the next performed audit if the problems are solved or the functioning of subprocesses have improved. Within the COSO framework, discussed in Chapter 2, this is called "effectiveness and efficiency of operations".

A use case diagram for this scenario is provided in Figure 5.2.

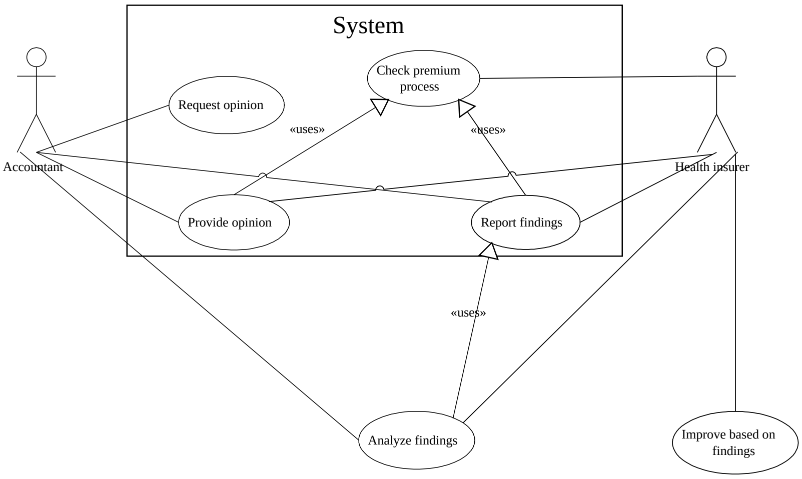

Figure 5.3: Use case "Remediation"

5.2.2 Functional requirements

This subsection lists the functional user requirements. The use cases in the previous subsection are used to illustrate basic functionalities during discussions, from which the requirements follow.

Process

The process executed by the system, should meet the following requirements:

UP-RQ1[1] The process checks the premium process to form an opinion on completeness, accuracy, and validity of the items in the accounts related to the premium process.

UP-RQ2 The process checks restricted access to the applications part of the premium process.

UP-RQ3 The process checks if the premium process is compliant with law and regulations.

UP-RQ4 The process performs a minimal amount of checks on the premium process, if the health insurer has a high level of internal control.

UP-RQ5 The process performs a maximal amount of checks on the premium process, if the health insurer has a low level of internal control.

System

The system itself should meet the following requirements:

US-RQ1 The system is configured based on the needs and the level of internal control of the health insurer.

US-RQ2 The system operates on data of the premium process that can be provided digitally and automatically by the health insurer.

US-RQ3 The system delivers an opinion and a report with findings.

Data

The data used or delivered by the system, should meet the following requirements:

UD-RQ1 The data of the premium process used by the audit software system is provided digitally and automatically.

UD-RQ2 The data of the premium process is provided by the health insurer, to achieve this, the health insurer's systems are adapted minimally.

UD-RQ3 The data of the premium process is offered to the audit software system in the format the premium process uses.

[1] The format of numbering requirements throughout this report is: U or S for user or system requirements, P, S, D, O or N for process, system, data, organization or non-functional and RQ for requirement

UD-RQ4 The opinion issued by the audit software system is trustworthy, it is hard to forge the provided proof containing the opinion.

Organization

The organization viewpoint should include the following requirements:

UO-RQ1 The accountant is responsible for keeping the system in running order.
UO-RQ2 The accountant is responsible for configuring the system.
UO-RQ3 The accountant is responsible for the opinion provided by the system.
UO-RQ4 The accountant is responsible for the communication between the premium process and audit software system.
UO-RQ5 The health insurer is responsible for making the requested data available.

5.2.3 Non-functional requirements

The following non-functional requirements are specified:

UN-RQ1 The connection between the premium process and the audit software system is secure.
UN-RQ2 The data of the premium process used by the audit software system is treated confidentially.
UN-RQ3 The audit software system is reliable and its reliability is checked regularly.

5.3 System requirements

This section is concerned with the requirements the system should meet. The system requirements have been elicited from interviews with accountants and IT-auditors from PwC, from interviews with employees at a health insurer, from the book “Bestuurlijke informatieverzorging 2B/toepassingen” [SMJ98] and from the user requirements cited in Section 5.2.

The goal of the system and the system’s stakeholders are given in Section 5.2 on user requirements.

5.3.1 Functional requirements

Process

The process executed by the system, should meet the following requirements:

SP-RQ1 The process converts data of the premium process to a usable format.

SP-RQ2 The process distinguishes operational data, containing operational input and output of the premium process, and process data, containing information on the execution of subprocesses of the premium process.

SP-RQ3 The process checks both operational data and process data to form an opinion on completeness, accuracy, and validity of the items in the accounts related to the premium process and restricted access to the applications part of the premium process, or on compliance.

SP-RQ4 The process knows the business rules related to the premium process and checks if the premium process proceeds conforming these rules.

SP-RQ5 The process performs process-oriented checks, it does not analyze data statistically.

SP-RQ6 The process performs a minimal amount of checks on the premium process, if the health insurer has a high level of internal control.

SP-RQ7 The process performs a maximal amount of checks on the premium process, if the health insurer has a low level of internal control.

SP-RQ8 The process combines the findings on all checks to form an overall opinion.

SP-RQ9 The process reports the opinion and findings.

System

The system itself should meet the following requirements:

SS-RQ1 The system receives the data from the health insurer automatically and digitally.

SS-RQ2 The system can handle the receipt of data both in a deferred or real-time fashion.

SS-RQ3 The system operates on operational data and process data, obtained from the health insurer.

SS-RQ4 The system uses a configuration, based on the needs and the level of internal control of the health insurer.

SS-RQ5 The system uses management data of the health insurer which contains business rules.

SS-RQ6 The system reports the opinion and findings to the accountant.

SS-RQ7 The system reports the opinion and findings directly and automatically to the health insurer.

Data

The data used by or delivered by the system, should meet the following requirements:

SD-RQ1 The operational data and process data of the premium process used by the audit software system is provided digitally and automatically.

SD-RQ2 The management data of the health insurer related to the premium process and used by the audit software system, is provided digitally and automatically.

SD-RQ3 The data of the premium process is provided by the health insurer, to achieve this, the health insurer's systems are adapted minimally.

SD-RQ4 The operational data and process data is offered to the audit software system in the format the premium process uses.
SD-RQ5 The opinion issued by the audit software system is trustworthy; it is hard to forge the provided proof containing the opinion.

Organization

The organization viewpoint should include the following requirements:

SO-RQ1 The accountant is responsible for keeping the system in running order.
SO-RQ2 The accountant is responsible for configuring the system.
SO-RQ3 The accountant is responsible for the opinion provided by the system.
SO-RQ4 The health insurer is responsible for making the requested data available.

5.3.2 Non-functional requirements

The following non-functional requirements are specified:

SN-RQ1 The system is adaptable to changing laws and regulations.
SN-RQ2 The system is adaptable to a changing format of data from the premium process which is input for the audit system.
SN-RQ3 The system is adaptable to a different output format of the opinion or report with findings.
SN-RQ4 The system is adaptable to new checks to be performed with an audit.
SN-RQ5 The connection between the premium process and the audit software system is secure.
SN-RQ6 The data of the premium process used by the audit software system is treated confidentially.
SN-RQ7 The system is reliable and its reliability is assessed by a third party, providing a certificate on reliability.

The requirements provided in this chapter, form the foundation for the functional description in Chapter 6, the data and organization viewpoint in Chapter 7, and the system viewpoint in Chapter 8.

Chapter 6

Future situation — Functional description

This chapter gives a functional description and the process viewpoint of the audit system for IT-enabled, continuous auditing, as explained in Chapter 3. By the audit system we mean the audit system in the broadest sense; the software system at the operational level but also the manual procedures associated with the software system on tactical and strategic level. In the remainder of this thesis, we refer to the audit system for IT-enabled, continuous auditing simply as 'the audit system'. Section 6.1 gives a high level overview of the collaboration between the health insurer and the accountant. The audit system is designed to be flexible: various system configurations can be used to set up the audit. The configurations are determined at the tactical level, a description is provided in Section 6.2. The operational level of the audit system is described in Section 6.3, and further elaborated in Section 6.4.

6.1 Collaboration health insurer and accountant

Before going into detail of the audit system, we give a high level overview of the system and the connection with the premium process in Figure 6.1. The modeling technique used throughout this chapter, is the Data Flow Diagramming technique explained in Appendix A.

At the strategic level, the management of both companies agree on a long term collaboration. Some adjustments need to be carried out at the operational level to make cooperation of the systems possible. Decisions on the implementation of the adjustments are made at the tactical level and prompted by the collaboration at the strategic level. The agreement on collaboration is internally communicated at the accountant's side by using Service Level Agreements (SLAs), these are formal negotiated agreements on the (quality) level of services. An SLA can include a definition of services, performance measurements, customer duties, problem management, and termination of agreement.

At the tactical level, the management of the health insurer orders the accountant to audit the premium process and the audit system is configured for the order. This is discussed in

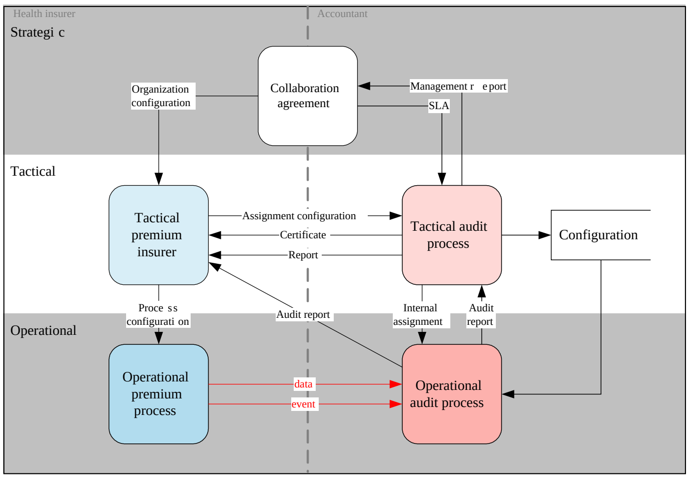

Figure 6.1: Relation premium process and audit system

Section 6.2. The audit is performed on data and process data, as described in Section 6.3.2. At the operational level, data and process data (event log files) are sent from the premium process to the audit system. Subsequently, the audit is performed and the results of the audit are sent directly to the health insurer or to the tactical management on the accountant's side. The accountant converts, checks and signs the report and certificate and sends it to the health insurer. With sending results directly to the health insurer, the audit system is moving further towards continuous auditing mentioned in Chapter 2. Important findings are also reported to the top management of the health insurer at the strategic level, since the management has final responsibility of the premium process, and needs to have up to date information.

Section 6.3 discusses the operational audit process in detail. To formalize the process viewpoint, a process model in the form of an activity diagram of the collaboration of the health insurer and the accountant, is given in Appendix C.

6.2 Configuration of the audit system

Following from the requirements in Section 5.3, the audit system needs to be adjustable to changing regulations, changing processes within the health insurer and changing audit assignments. These changes can occur regularly, thus the audit system should be able to easily adapt to changes. Therefore we decided to design a flexible system with various configurations for various audit assignments. As shown in Figure 6.1, the preconditions for the audit are filled in at the tactical level, they are mostly not standardized or digitally stored. Therefore, the configuration is entered manually. An overview is given in Figure 6.2.

When the health insurer orders the accountant to perform an audit, they first need to agree on the scope of the audit, e.g. the period to be audited, what kind of audit to perform and

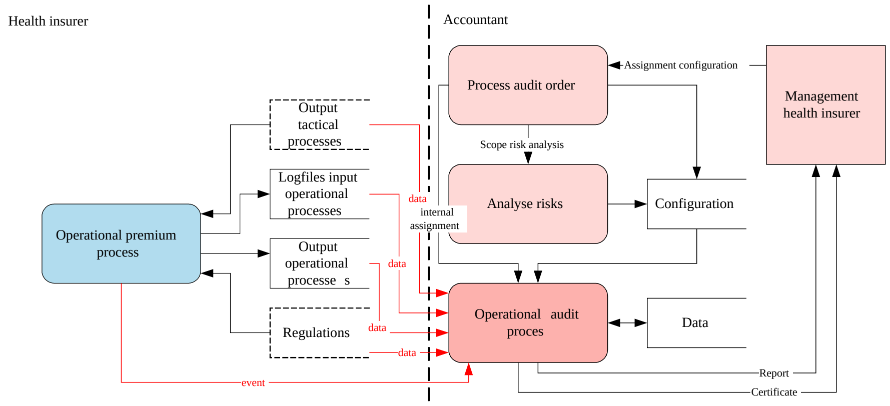

Figure 6.2: Tactical audit process and relation with premium process

what form the output of the audit will have. The manual process "Process audit order" in Figure 6.2 handles the scope of the audit and translates it to input for the configuration.

With the traditional audit, the basis of the audit is a risk analysis. Systems, processes and transactions are evaluated to get insights in potential risks. Data produced by processes with higher potential risks are analyzed profoundly. Besides checking systems, processes and transactions, also the level of internal control is determined. If a health insurer has a high level of internal control, risks of fraud and errors with a financial repercussion are low. The internal control activities and environment, when monitored and communicated clearly, following the COSO model [COS04], make sure errors are detected early in order to correct them. A low level of internal control implies a high risk, and a more extensive audit is performed. As shown in Figure 6.2, the audit system of the future uses the same risk analysis to determine which data and processes need to be checked and which checks, mentioned in Section 6.4, should be executed. These settings are also part of the configuration and thus adapted when necessary. Since risk analysis goes beyond systems and processes, and involves manual procedures and how people follow rules, it cannot be performed automatically. To assure reliability of the outcome of the audit performed by the audit system, a risk analysis should be executed with a regularity determined by the accountant. A change in risks leads to a new configuration of the audit system.

Regulations and the output of tactical processes of the health insurer, shown in Figure 6.2, provide information on the setup of the premium process. Therefore, regulations and tactical process output are also sent to the audit system. Section 6.3.2 goes more deeply into this.

For the process viewpoint, a process model of managing the audit order on tactical level, is given in Appendix C.

6.3 Operational audit process

This section describes the operational audit process and its relation to the operational premium process. For this purpose, the premium process is recalled and slightly adapted in Section 6.3.1. Section 6.3.2 explains the operational audit process.

6.3.1 Overview premium process

In Chapter 4 a functional description of the present situation of health insurers has been given. Recall that we decided to focus on the premium process for developing the audit system. Within the premium process, three sub processes are of concern for the financial audit; namely “Calculate premium”, “Subscribe/Mutate/Unsubscribe”, and “Financially manage premiums”. To keep a clear overview of this simplified premium process, the incoming management data (output of tactical processes and regulations) and outgoing operational data of the premium process that is stored for usage by the audit process, are aggregated as shown in Figure 6.3. The data is still separately stored when the process is viewed at a lower abstraction level. Raw input of the premium process, e.g. subscriptions, is logged and stored for use by the audit system.

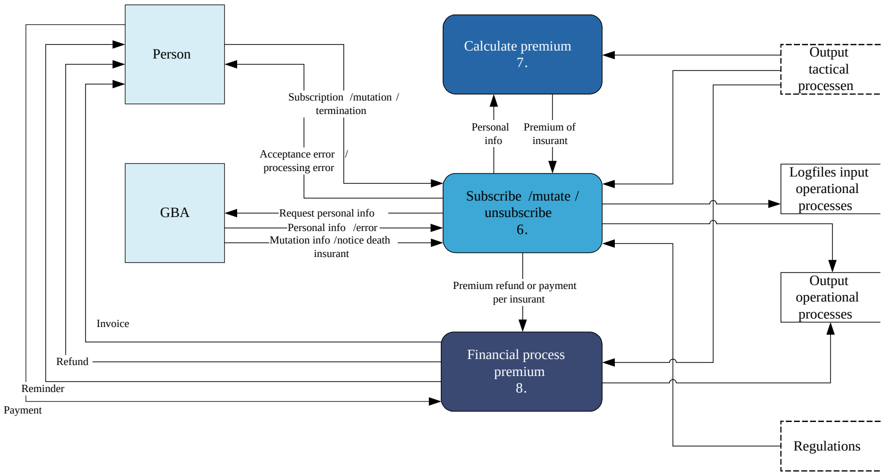
Person
GBA
Calculate premium 7.
Output tactical processen
Subscription /mutation / termination
Acceptance error / processing error
Personal info
Premium of insurant
Subscribe /mutate / unsubscribe 6.
Logfiles input operational processes
Request personal info
Personal info /error
Mutation info /notice death insurant
Premium refund or payment per insurant
Output operational processes
Invoice
Refund
Reminder
Payment
Financial process premium 8.
Regulations

Figure 6.3: Premium process with tactical input and operational output aggregated

6.3.2 Overview audit system

Chapter 5 on requirements states that the goal of the audit system is to provide an independent opinion on the completeness, accuracy, and validity of items in the accounts related to the premium process, restricted access to financial systems used by the premium process and an independent opinion on the compliance of the premium process with law and regulations. This gives rise to the following questions: In what form is the independent opinion provided, what data is checked to come to this opinion, how is this data collected and how is it checked? This section provides an overview of the operational audit process and discusses the decisions made concerning these questions.

Figure 6.4 gives a high level overview of the operational audit process and its connection with the premium process, discussed in Section 6.3.1. An overview of the connection between the audit system and the elaborated premium process, is given in Appendix D, to show the specific data that is collected.

Following from the requirements in Section 5.2, the independent opinion should be provided automatically and digitally and should be trustworthy. Therefore we have chosen for providing the opinion in the form of digital certificates with digital signatures. Section 8.2.1 discusses this subject in more detail.

For the audit system to form an opinion on completeness, accuracy, validity, restricted access and compliance, data that can be retrieved automatically should be examined. This is the case with digitally stored data that is input and output of the premium process and process information of the sub processes. With a traditional audit, the data is asked from the health insurer and the process information is provided by employees showing how processes work and analyzing how cases find their way through the process. In the audit system, the data and process information are automatically collected through a connection between the health insurer's premium process and the audit system. Process information is stored in the form of event log files, which contain information on e.g. the name of the process, the execution time of a process, the status of the process and the person who executed the process. The data and event log files are received from the health insurer and converted to the format used by the audit system, as shown in Figure 6.4. Section 8.3 explains more about the format of sent data by the health insurer.

Figure 6.3 shows that the premium process has two types of data as input and output: management data and operational data. Operational data is used and produced by the operational premium process, and includes e.g. a subscription form or the information of a new insurant. Management data consists of regulations and output of tactical processes and provides input for the setup of the operational premium process. Examples of management data include a list of authorized people for executing a specific process and a period after which a reminder of payment is sent. Operational data is subject to the checks of the audit system and management data is used to configure the audit system, as mentioned in Section 6.2. Therefore, both are sent to the audit system, shown in Figure 6.4 and further discussed in Section 6.4.1. Operational data that is input for the premium process, e.g. GBA information, is also stored so that it can be used by the audit system.

Data and event log files can be collected in a real-time or deferred fashion. Collecting real-time means that every time data is changed or an event has taken place, the data or the specific line from the event log file, containing process information on the event, is sent to the accountant and stored on the accountant's side. With deferred collection, data and event log files are stored on the insurer's side and sent to the accountant only at specific points in time. When collecting data in a deferred fashion, an order to pick up the data is sent to the "Receive and convert data" process. This is indicated with the data flow "Order*" in Figure 6.4. With real-time data collection, this order does not exist. Following from the

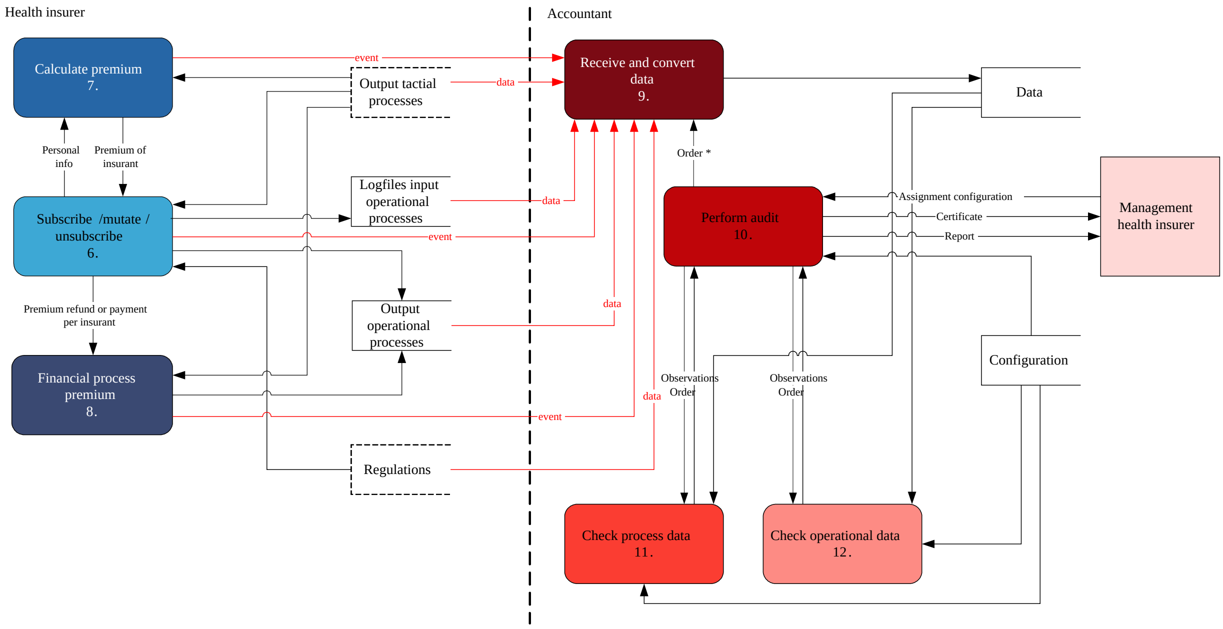

Figure 6.4: Operational audit process and relation premium process

requirements in Section 5.3, the audit system should be flexible. Therefore the audit system supports both ways of data collection. Which way of data collection to use, can be decided in consultation with the health insurer and is recorded in the configuration. More on real-time and deferred data collection is given in Section 6.4.1.

Because of the different format of event log files and operational data, a distinction is made in checking event log files (process data) and operational data. The processes “Check process data” and “Checking operational data” in Figure 6.4 have a different input format. Detailed information on the checks of process and operational data, is given in the next section.

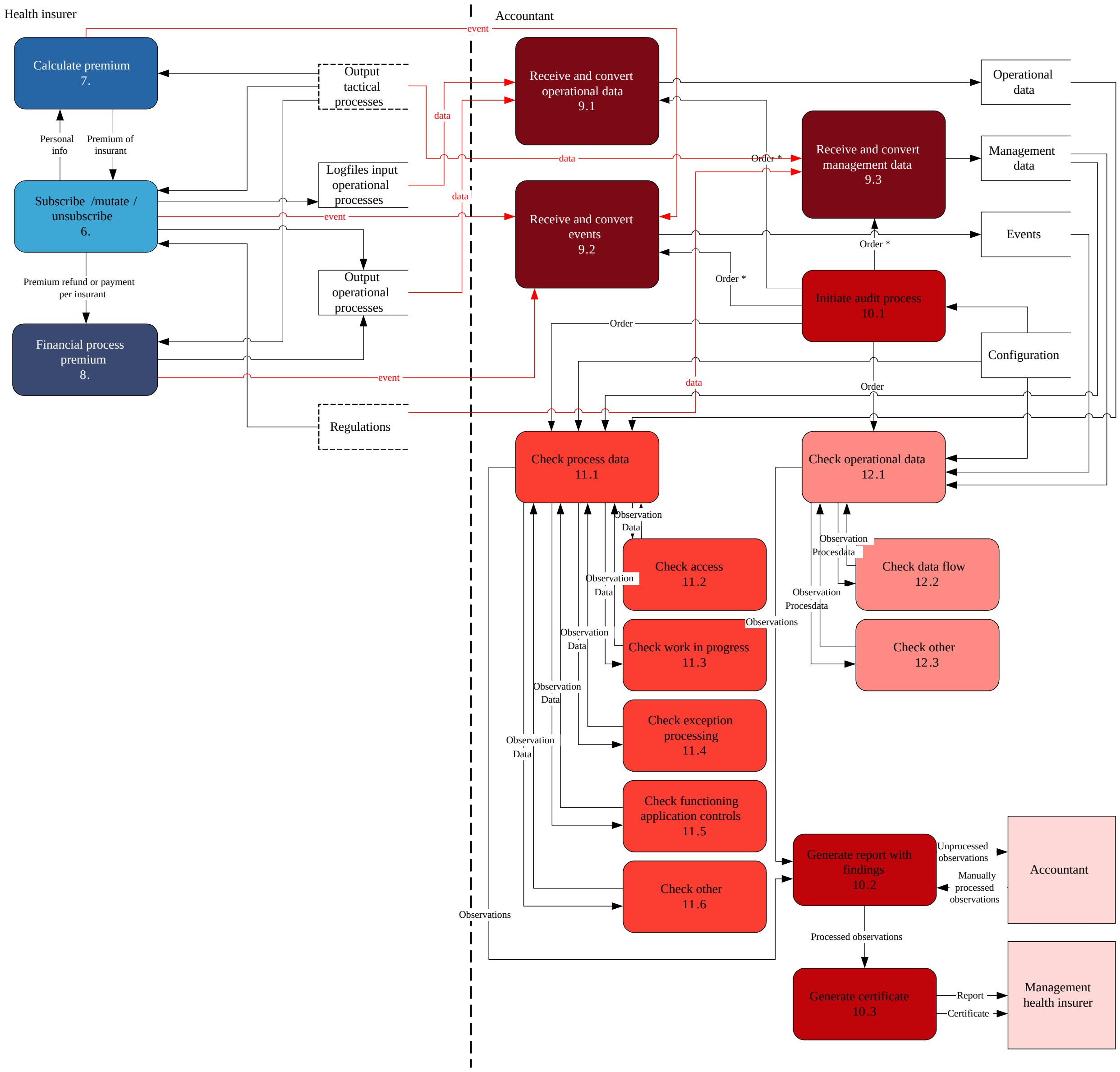

Figure 6.5: Operational audit process in detail and relation premium process

6.4 Elaborated operational audit process

The previous section explains the operational level of the audit system. The high level overview of the operational audit process given in Figure 6.4, is further elaborated in Figure 6.5. The four main sub processes “Receive and convert data”, “Check operational data”, “Check process data”, and “Perform audit” are further specified. This section goes more deeply into these processes.

6.4.1 Receive and convert data

The process “Receive and convert data” is further divided into “Receive and convert operational data”, “Receive and convert events”, and “Receive and convert management data” as shown in Figure 6.5. The different types of data have been discussed in Section 6.3.2.

Section 6.3.2 also mentions briefly the difference between collecting data real-time or deferred. Real-time data collection implies that changing data or information on executed events in the form of lines from event logs, are continuously sent to and stored on the accountant’s side. Deferred data collection involves data and event logs to be stored on the insurer’s side and to be sent to the accountant at specific points in time. UML activity diagrams are used to model the difference between real-time and deferred data collection within the audit system. The activity diagrams can be found in Appendix C.

6.4.2 Check operational and process data

As mentioned in Section 6.3.2, operational data and process data are checked to form an independent opinion on completeness, accuracy, validity, restricted access, and compliance of the premium process. Since it is not straightforward to define these characteristics precisely, a different framework on financial statement assertions, matching the mentioned characteristics, has been chosen. The American Institute of Certified Public Accountants’ Statement provides the framework “Auditing Standards No. 106: Audit Evidence” [AIC07] and describes the use of assertions in audit procedures. Since we are interested in auditing the health insurer’s systems that underlie and therefore impact the financial statements, and not interested in account balances or presentation and disclosure, we particularly focus on the assertions about classes of transactions and events for the period under audit [AIC07]:

Occurrence Transactions and events that have been recorded have occurred and pertain to the entity

Completeness All transactions and events that should have been recorded have been recorded

Accuracy Amounts and other data relating to recorded transactions and events have been recorded appropriately

Cutoff Transactions and events have been recorded in the correct accounting period

Classification Transactions and events have been recorded in the proper accounts

This list can be extended in the future with assertions of different categories.

In consultation with accountants from PwC we decided that the following checks on operational data and processes of the health insurer should be realized to cover the chosen assertions. Figure 6.5 gives an overview of these checks included as functional modules in the audit system. "Check other" emphasizes the possibility of extending the modules.

Check process data	**Check operational data**
Check access	Check data flow
Check work in progress	Check other
Check exception processing	
Check functioning application controls	
Check other	

These checks are explained briefly:

Check data flow The data flow between different systems, applications and to the general ledger is checked on correctness

Check access Access to systems and applications of authorized people is checked

Check work in progress The amount of work, of cases waiting to be processed by the system and applications, is checked

Check exception processing Cases that are not accepted by the system or rejected during processing, are checked on being handled correctly

Check functioning application controls Application controls are checked on being operational, using the correct parameters provided by the health insurer (mentioned in Section 6.3.2 as management data)

Check other This module has no functionality but can be replaced by new functional modules, it is added to underline the extensiveness of modules

The relation between the assertions and the coverage by functional modules is given in the following matrix. Note that this is not a one to one relation; checking each assertion is done by several modules. As a consequence, each module is associated with multiple assertions.

Assertion **Module**	*Occurrence*	*Completeness*	*Accuracy*	*Cutoff*	*Classification*
Check operational data					
Check data flow	x	x	x	x	x
Check process data					
Check access	x	x			x
Check work in progress		x		x	
Check exception processing	x	x	x		
Check functioning application controls	x		x	x	x

By way of illustration, we explain how the functional modules are used to check the assertions. The objective is not to give a complete overview, but to give a feeling on how the functional modules are applied.

Occurrence

- Check data flow to ensure that data from one application is received by the next application.
- Check access so no one unauthorized could have added cases.
- Check exception processing for assuring that unaccepted cases are rejected by the system.
- Check functioning application controls for assuring that accepted cases are not rejected by the system.

Completeness

- Check data flow to ensure that all data from one application is received by the next application.
- Check access so no one unauthorized could have deleted cases.
- Check work in progress to make sure cases are eventually handled and processed.
- Check exception processing for assuring that unaccepted cases are rejected by the system.

Accuracy

- Check data flow to ensure that data from one application is received correctly by the next application.
- Check exception processing for assuring that unaccepted cases are rejected by the system and not justified incorrectly.
- Check functioning application controls for assuring that accepted cases are not rejected by the system.

Cutoff

- Check data flow to ensure that data from one application is received by the next application within the correct time period.
- Check work in progress to make sure cases are eventually processed.
- Check functioning application controls to assure that cases are handled within the correct time period.

Classification

- Check data flow to ensure that data from one application is received correctly by the next application.
- Check access so no one unauthorized could have changed cases.
- Check functioning application controls to assure that cases are handled having relations with the proper accounts.

Each functional module can check each process, although it is also possible to leave processes out. The configuration contains which module checks which process, this setup is based on the risk analysis and order of the health insurer discussed in Section 6.2.

6.4.3 Perform audit

The process “Perform audit” is composed of “Initialize audit”, “Generate report with findings”, and “Generate certificate”. The initialization activates the audit and triggers the receipt of data in the case of deferred data collection. After checking the operational and process data, the results are combined and reported. When it is not possible to combine and report results automatically, the accountant processes the results manually. After the report is generated, a certificate is produced and signed. Chapter 8 discusses possible technologies used to this end. Chapter 7 provides a structural overview of data used by the audit system, including the certificate.

Chapter 7

Future situation — Data and organization viewpoint

The previous chapter was concerned with a functional description of the audit system. This chapter goes more deeply into the corresponding data viewpoint, see Section 7.1, and organization viewpoint, see Section 7.2. The data and organizational viewpoint are part of the architectural framework subject to this research, defined in Chapter 3.

7.1 Data viewpoint

The data viewpoint provides models of the data produced and consumed by the audit system on the operational level. A data viewpoint is used for the development of an information system and shows the structure of data that is used or produced by a business process or application. Furthermore, a data viewpoint may provide a representation of data at the business level, or data at the application level [LTP$^+$05].

Modeling all produced and consumed data would be a time consuming task and not very meaningful in the light of this research. Therefore we decided to model four different kinds of data. This can be extended in the future; the four models described in this section can serve as examples for new models. The four kinds of data include traditional input, new output, internally used and produced data and the configuration. For modeling traditional input, we have chosen to model information on the insurant, similar to information currently used by an audit. For the output we have chosen the certificate, because a new audit certificate, different from the traditional audit certificate, is designed for the audit system. As for internal data, observations are modeled. The configuration is chosen due to its importance for the audit system. See Figure 6.5 in Chapter 6 for the use of information on insurants, certificates, observations, and configurations by the audit system.

For modeling data, UML class diagrams are used. A short description of UML class diagrams is provided in Appendix B. Data can be modeled at three levels: conceptual, logical and physical [SW04]. The conceptual data model is a technology independent specification of the data, used to explore the domain and for communication between the data modeler and stakeholders. The logical data model is a translation of the conceptual data model into structures to be implemented with a particular data management technology, e.g. relational tables and columns, object-oriented classes, or XML tags. The physical data model extends the logical level by providing a specification of the physical storage and access mechanisms. The goal of this research is to provide a high level design of the audit system, which can be further explored and extended for a future implementation. However, this extension is outside the scope of this research, and so are logical and physical data models. Therefore, this chapter focuses on conceptual data modeling.

7.1.1 Insurant

Information on the insurant is stored in the insurant data store, shown in Figure 4.5 in Chapter 4. A data model of this information is provided in Figure 7.1. An insurant has some general information as name and address. He also has one or more insurance agreements, one for each year the insurant was insured. An insurant could have made several payments of the premium, and could have received a refund if the insurance is terminated during a period already paid. Payments and remittances (containing also refunds) are related to the insurance agreement.

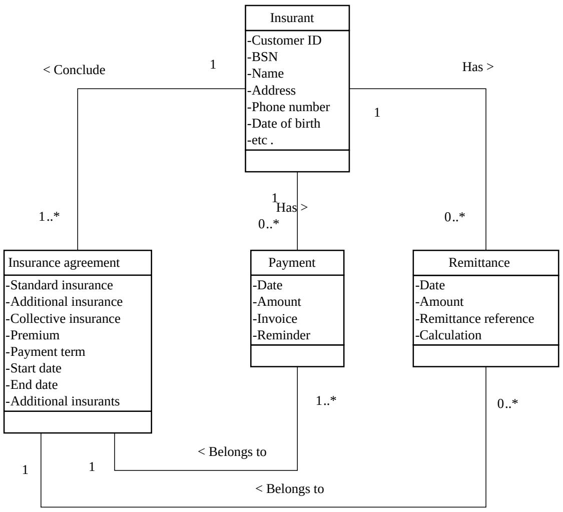

Figure 7.1: Data model of insurant

7.1.2 Configuration

Chapter 6 on the functional description of the audit system, introduces the idea of keeping the audit system flexible by using different configurations. The data model of a configuration is depicted in Figure 7.2. A configuration consists of input settings, output settings and checked processes. Input settings are e.g. which population of data should be subject of the audit, or if data should be collected in a deferred or real time fashion. Output settings include which level of detail the output should have, or after which period of time output should be generated. Some of these settings have to be included in the certificate, therefore each setting has a parameter “In certificate” which can be true, to be included, or false, to be excluded in the certificate. This is explained in more detail in Section 7.1.4.

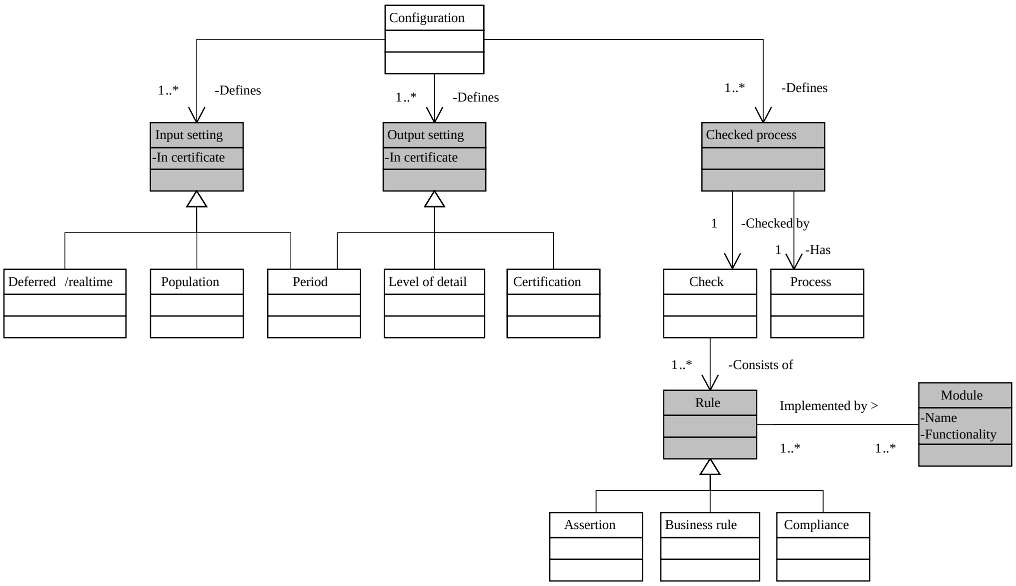

Figure 7.2: Data model of configuration

A “Checked process” consists of one process with one check applied to it. Section 6.2 discussed configuring the audit system, it mentioned that not every process needs to be checked, dependent on the level of internal control of the health insurer. To indicate the scope of the audit in the report with findings, the checked processes with their corresponding checks need to be known. Therefore a separate class is introduced for checked processes.

Checks, performed on processes, consist of rules. A rule can be an assertion, discussed in Section 6.4, a business rule extracted from management data, e.g. refunds higher than 1000 euro should be approved manually, or a compliance rule, dependent on current regulations. For the financial audit, assertions are used. Compliance rules are used to give an opinion on compliance. Business rules are only used for internal use for the health insurer, to improve efficiency and effectiveness of operations, mentioned by the COSO framework discussed in Section 2.2. Section 6.4 explains the implementation of assertions by functional modules. Functional modules can also be used for the implementation of business rules or compliance rules. For reasons of complexity, we do not want to go too deeply into implementation issues for the design of the audit system, we leave the relation between modules and business rules or compliance rules out. The matrix containing the relation between the modules and assertions has been given in Section 6.4. A representation of this relation, between the rule and module, is given in Figure 7.2. The grey classes in the model represent classes which are also used in other data models. Input and output settings, rules and check processes are used in the data model of the certificate, discussed in Section 7.1.4. Rules and modules are used in the model of observations, explained in the next subsection.

7.1.3 Observation

For representing internal data of the audit system, observations are modeled. An overview is given in Figure 7.3. A functional module, or module for short, checks operational data or process data as mentioned in Section 6.4, and gives an opinion as a result. Checked data is included in the model as a set with certain properties instead of a set of separate facts, and is therefore called "Data specification". The previous subsection discusses the relation between modules and rules, these classes are also part of a configuration. Rules are included in the observation data model, to express the connection between an observation and a configuration.

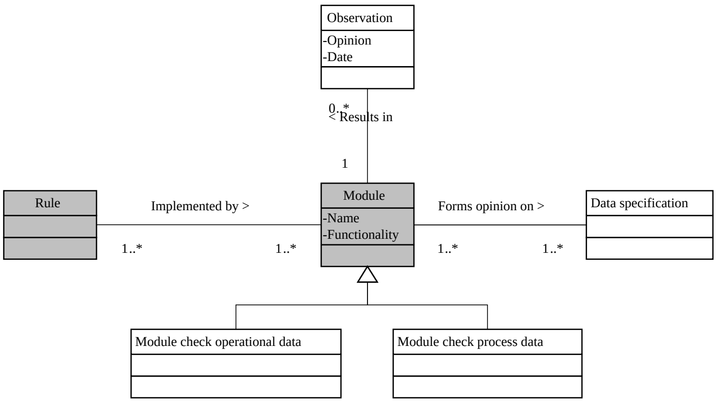

Figure 7.3: Data model of observation

7.1.4 Certificate

Following the requirements in Section 5.3, the output of the audit system is a certificate that contains an opinion on completeness, accuracy, and validity of items in the accounts related to the premium cycle and on restricted access to financial systems used by the premium cycle and/or on compliance with laws and regulations. An overview is given in Figure 7.4. Section 6.4 discusses the use of assertions instead of completeness, accuracy, validity and restricted access. In the light of the new, IT-enabled audit, it is possible to extend the traditional audit certificate, having one general opinion, with clauses containing an opinion on the checked assertions and compliance rules. It is also possible to include which processes are checked, useful for e.g. future reference or further investigation. The use of style sheets, discussed in Section 8.2.1 in the next chapter containing the system viewpoint, makes it feasible to provide a different view of the certificate, with or without certain clauses, to each stakeholder.

Besides containing the opinion, the certificate also includes a header and footer. The header contains information on the assignment, e.g. the date of the certificate, the company which is audited, and the subject of the audit. The scope of the audit is also inserted in the header, therefore input and output settings defined in the configuration and of interest for the certificate are included. With "Of interest", all settings with the constraint "In certificate = true" are meant, explained in Section 7.1.2. The footer contains an electronic certificate

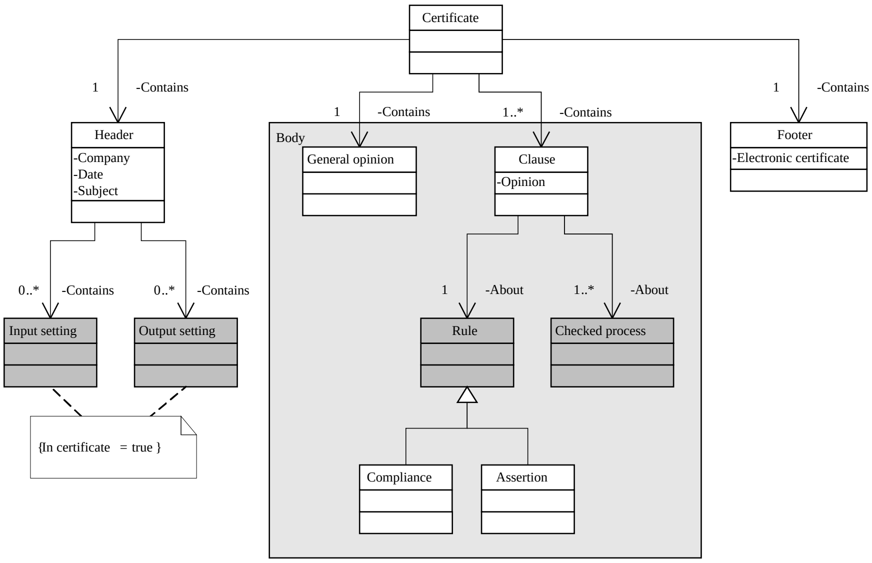

Figure 7.4: Data model of certificate

which replaces the accountants signature in traditional audit certificates. This means that the audit system outputs a certificate containing an opinion and an electronic certificate. An electronic certificate verifies the sender's, in this case the accountant's, identity, which can be compared to a passport [SB02]. More on electronic certificates is found in Section 8.2.1 of the next chapter.

7.2 Organization viewpoint

The organization viewpoint shows the structure of the internal organization of an enterprise, department, or other organizational entity [LTP$^+$05]. The organization viewpoint is used to identify authority, competences and responsibilities within an organization. This section describes the organization viewpoint of the complete audit system, including tactical and operational processes as mentioned in Chapter 6, and the premium process together, since they are closely interrelated. Recall the overview of the relation between the health insurer and the accountant, given in Figure 6.1. A distinction is made between collaboration at the strategic level, at the tactical level, and at the operational level. To refine responsibilities at the strategic level, the process "Agree on collaboration" in Figure 6.1 is divided into three processes: "Order for audit", "Compose audit approach", and "Negotiate on collaboration".

Figure 7.5 shows the division into responsibilities of the premium process and the audit system, with "Agree on collaboration" refined as mentioned above, and the six matching actors. At each level, two actors are distinguished, one on the health insurer's side and the other on the accountant's side.

At the strategic level, the actor on the health insurer's side (***1***) is responsible for ordering the accountant to audit. This actor is generally speaking represented by the executive board

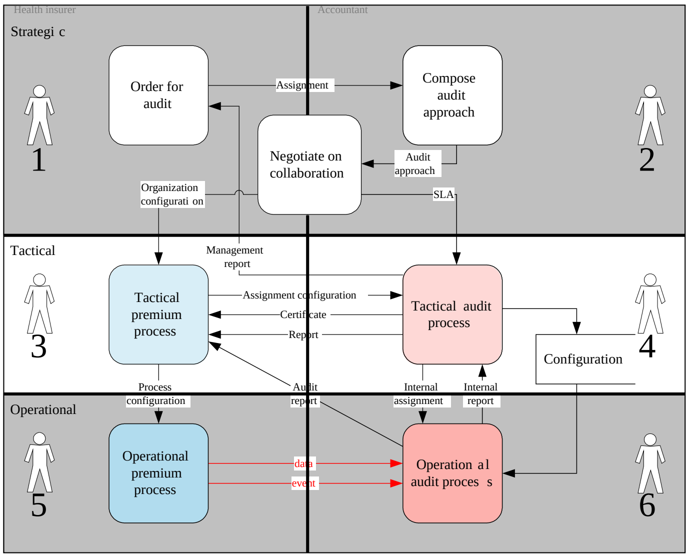

Figure 7.5: Organization of premium and audit process and responsible actors

of the health insurer. Next, the actor on the accountant's side (***2***), represented one or more partners of the accounting firm, is responsible for composing an audit approach for the order. Both the health insurer and the accountant (***1*** and ***2***) are responsible for negotiate the audit approach, to come to an agreement on collaboration. At the tactical level, both actors (***3*** and ***4***) are responsible for managing the tactical processes at their side. The actors represent the middle management of the companies. At the operational level, both actors (***5*** and ***6***), presumably managers of the staff carrying out the work, are responsible for executing the operational processes at their side. This means that a certified public accountant has final responsibility for the audit system. It is probable that a system expert is involved in the responsibility for the audit software system.

The flow of data between the health insurer and accountant, and internally is organized as follows:

Data flow:	**Responsible actor:**
Assignment	***1***
Audit approach	***2***
SLA	***2***
Organization configuration	***1***
Management report	***4***

Data flow:	**Responsible actor:**
Assignment configuration	***3*** is responsible for the explicit assignment when starting the audit; ***4*** is responsible for selecting preconditions if a financial or compliance audit is performed, see Section 3.3; ***3*** is responsible for selecting preconditions with the configuration if the audit concerns operational efficiency and effectiveness, see Section 3.3.
Certificate	***4***
Report	***4***
Audit report	***6***
Process configuration	***3***
Internal assignment	***4***
Internal report	***6***
Filling in configuration	***4***
Data and event	***5*** is responsible for storing data and events in the right format and location; ***6*** is responsible for the communication of data and events to the audit system.

An accounting firm can be classified as a professional bureaucracy according to Mintzberg [Min83]. Within a professional bureaucracy, technical activities are performed by highly skilled workers since the nature of work is complex. As a result, the control is decentralized and depends on universal professional standards, defined by professional associations as the NIVRA (Dutch association for certified public accountants). Typical for professional bureaucracies is the rigid structure, which makes it hard to adapt to the production of new output. Besides, making use of current technologies to support highly skilled workers in their daily job, reduces the workers' autonomy and encounters opposition [Min83]. This is the reason why certified public accountants strongly resist accepting final responsibility for the audit system, they argue not to be educated in system management. Changes in the profession of accountants are only accepted when carried out step-by-step, starting with introducing the changes into the education [Min83].

Chapter 8

Future situation — System viewpoint

This chapter provides the system viewpoint of the audit system, for which Chapter 6 has given the functional description and Chapter 7 the data and organization viewpoint. In literature, the system viewpoint is named a system architecture. Therefore, this chapter refers to the system architecture instead of the system viewpoint. Section 8.1 gives a general introduction to system architectures and architectural patterns. The system architecture of audit system is divided into two layers, described in Section 8.2. The communication between the audit system and the premium process is discussed in Section 8.3.

8.1 System architecture and architectural patterns

The previous chapters have provided the functional description, the data viewpoint and the organization viewpoint of the audit system. This chapter provides the system architecture (system viewpoint) of the audit system. The Software Engineering Institute from Carnegie Mellon University provides the following definition of a system architecture [SEI07]:

> "A representation of a system in which there is a mapping of functionality onto hardware and software components, a mapping of the software architecture onto the hardware architecture, and human interaction with these components."

The system architecture provided in this chapter, serves as a blueprint for the implementation of the designed audit system. Because the starting point for the design of the audit system is a future situation ten years from now, the hardware architecture is left out, since we do not know what hardware will be available ten years from now. In addition, emphasis within the design of the audit system is on functionality captured by software, not on hardware. For the software architecture, we provide general guidelines and discuss problems we already foresee. Providing a complete software architecture would be too complex and time consuming, and would outrun the objective of this research, since the objective is only to investigate the possibilities of IT-enabled, continuous auditing. Our focus is therefore on conceptual design; implementation issues are left out.

As stated in the definition above, the main idea is to map functionalities onto components. The easiest way to do this, is to reuse standard components. When elaborating the design of the audit system, standard components provide structure and transparency, and make it easy to reproduce the design. For standard components, architectural patterns are used. In "Pattern-oriented software architecture: A system of patterns", the following definition of architectural patterns is given [BMR$^+$96]:

> "An architectural pattern expresses a fundamental structural organization schema for software systems. It provides a set of predefined subsystems, specifies their responsibilities, and includes rules and guidelines for organizing the relationships between them."

Patterns are used as building blocks for designing a software system. A pattern provides a solution for a specific problem given a specific context. Avgeriou and Zdun discuss current issues with describing, finding and applying architectural patterns; this is still ad-hoc and unsystematic due to lacking semantic consensus on the representation of architectural patterns, due to confusion on the granularity of architectural patterns, and due to a missing classification or cataloguing of patterns [AZ05]. To overcome these issues, we assumed the following: as for the representation of architectural patterns we follow the definition of Buschmann et al. mentioned above[BMR$^+$96]; concerning granularity we put design patterns, which are medium scale architectural patterns concerning to Buschmann et al. [BMR$^+$96], in the same category as architectural patterns; and for a catalogue, we use "Pattern-oriented software architecture: A system of patterns" [BMR$^+$96] and "Architectural patterns revisited – A pattern language" [AZ05], unless stated otherwise.

As for the modeling technique used throughout this chapter, we use the same conventions as the book "Pattern-oriented software architecture: A system of patterns" [BMR$^+$96], with simple blocks for processing units and arrows for the communication between the units.

8.2 Two layer architecture

The designed audit system operates in a specific domain, namely the accountancy domain. Therefore a distinction between domain specific software and supporting software is desirable. In that way, it is easy to focus on the software most interesting for this research, without being dependent on the design of the supporting software. Therefore, a distinction is made into two different layers and the *layer* architectural pattern is applied [BMR$^+$96]. The two layers are the application layer, containing specific domain applications, and the middleware layer, responsible for the connection between the applications and the interaction of the applications on a network. A third layer could be distinguished for the infrastructure, consisting of the operating system and networking software. However, it is impossible to predict what operating systems and networking software will be available in ten years time, and we do not have specific recommendations for the infrastructure, therefore this layer is left out.

Figure 6.5 in Chapter 6 shows a data flow diagram of the audit system on operational level. This functional description is translated into the two layered system architecture. Requirements from Chapter 5 and applicable design decisions of the data and organization viewpoints are also taken into account, and referred to if necessary. The next subsections discuss the two layers of the system architecture, emphasizing the application layer since this layer contains the actual audit logic. For the middleware supporting the applications, some recommendations are given.

8.2.1 Application layer

The application layer consists of the applications representing the actual business logic. The operational level of the functional description, given in Section 6.4, defines the functionality of the audit system. This subsection provides the filling-in of the application level using architectural patterns based on the functionality of the audit system, see Figure 6.5. A future implementation, which is outside the scope of this thesis, can be based on the architectural design by patterns.

On a high level view of the audit system, with all data from the health insurer aggregated, the audit system has two kinds of input, see Figure 6.5: data sent by the health insurer, and the configuration defining the setup of the audit system. The data from the health insurer is checked and the findings are reported. The configuration defines how to check the data and how to report the findings. Two patterns are distinguished here: the *pipe and filter* architectural pattern for offering, handling (including checking) and reporting data and the *reflection* architectural pattern for configuring the audit system by means of the configuration [BMR+96]. The pipe and filter architecture is given in Figure 8.1 and the reflection architecture in Figure 8.2.

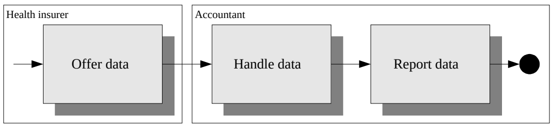

Figure 8.1: Data flow through applications

The fact that the audit system processes a stream of data, leads to the observation that the processing steps should be embedded in a *pipe and filter* pattern [BMR+96]. Each processing step is enclosed in a filter component and data is processed through pipes between filters. The black dot represents data sink for data storage. It is more likely that the reports are directly sent to another system, instead of being stored until the report is requested. For simplicity reasons, the audit system ends with generating and storing the report. Besides, even if the reports are sent to another system, it is desirable to save a copy within the audit system for future reference.

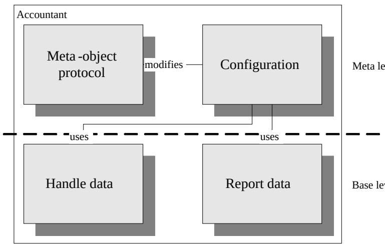

Figure 8.2: Use of the configuration

The configuration is introduced to keep the audit system flexible and adaptable to requirements that have changed, as required in Section 5.3. Consequently, the *reflection* pattern is most suitable for this situation [BMR+96]. The *reflection* pattern provides a mechanism for changing structure and behavior of systems dynamically: structural and behavioral aspects are stored into meta-objects and separated from the application logic components. Therefore, the *reflection* pattern can handle unforeseen changes in technology and requirements. Two implementation levels are introduced; the meta-level contains the meta-objects encapsulating and representing the structure and behavior of the software, and the base level contains the application logic components with their implementation depending on the meta-objects for independency of changes. A meta-object protocol is specified for changing the meta-objects and examining the correctness of a change specification.

As stated in Chapter 6 on the functional description of the audit system, the health insurer sends three kinds of data; operational data, process data and management data. Management data, as explained in Section 6.3.2, provides information on the design of the internal processes of the health insurer and is therefore used as input for the checking operational and process data. Similar to the configuration, management data changes over time. Therefore, a suitable solution for capturing the requirements set by the management data, is through the use of the *reflection* pattern, shown in Figure 8.3.

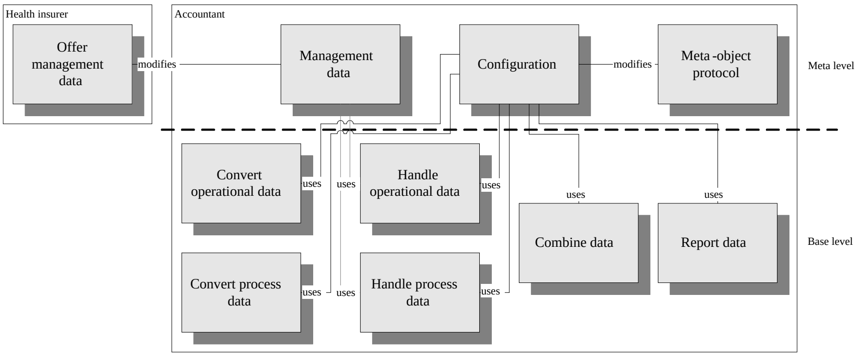

Figure 8.3: Use of management data

The operational data and process data are converted into the internal format of the audit system, before they are checked. After the examination of data, the findings are combined and reported as defined by the configuration. Figure 8.4 shows this process, it is obvious that the *pipe and filter* pattern also applies to this situation.

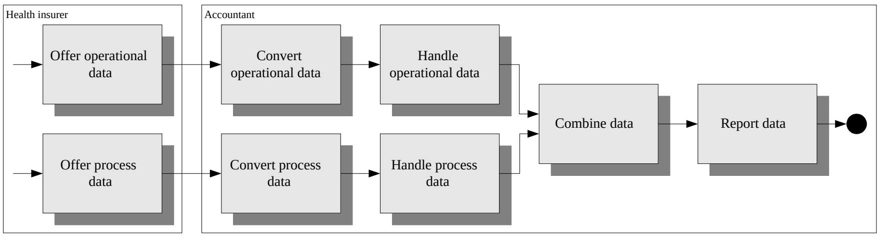

Figure 8.4: Data flow of operational and process data through applications

As mentioned above, the audit system should be kept flexible through the use of a configuration. Flexibility should also be found in the way the audit system handles functional modules. Add or remove functional modules for checking operational and process data should be simple. A way to achieve this is to see all functional modules as service components. Service components are defined by Tosic et al. as [TMP01]:

> “A self-contained, network accessible unit of service provisioning and management that encapsulates some service functionality and data, has a well-defined interface, and can be composed with other service components.”

We introduce a separate process to register what components are available and to invoke each service component with the correct data. The *client–server* architectural pattern serves this purpose [AZ05], the service components are represented by servers and the process registering and invoking services is represented by the client.

As already mentioned, the use of service components and the *client–server* pattern is appropriate for the implementation of functional modules. Furthermore, we also want to achieve flexibility in the way data is converted, since the audit system should be able to handle two kinds of data, offered in different formats. A format-change of data in the future should easily be handled. Another part of the audit system, suitable for an implementation by service components, is the reporting of findings. The functional description in Section 6.4 identifies two reporting functions, reporting all findings in a detailed fashion, and generating a certificate; other functions might be added in the future. Figure 8.5 gives the pipe and filter architecture from Figure 8.4 extended with the *client-server* pattern for converting data, check operational data, check process data, and report findings.

In Figure 8.5 the operational and process data sent by the health insurer are received together. For the audit system, it does not matter how the health insurer offers the data separately or combined, since the audit system detects what data is offered and chooses the right service component(s) to apply. The communication between the health insurer and the audit system is discussed in Section 8.3. User interfaces are not included in the architecture, since focus of the architecture is on the stream of (processed) data and not on the interaction with users.

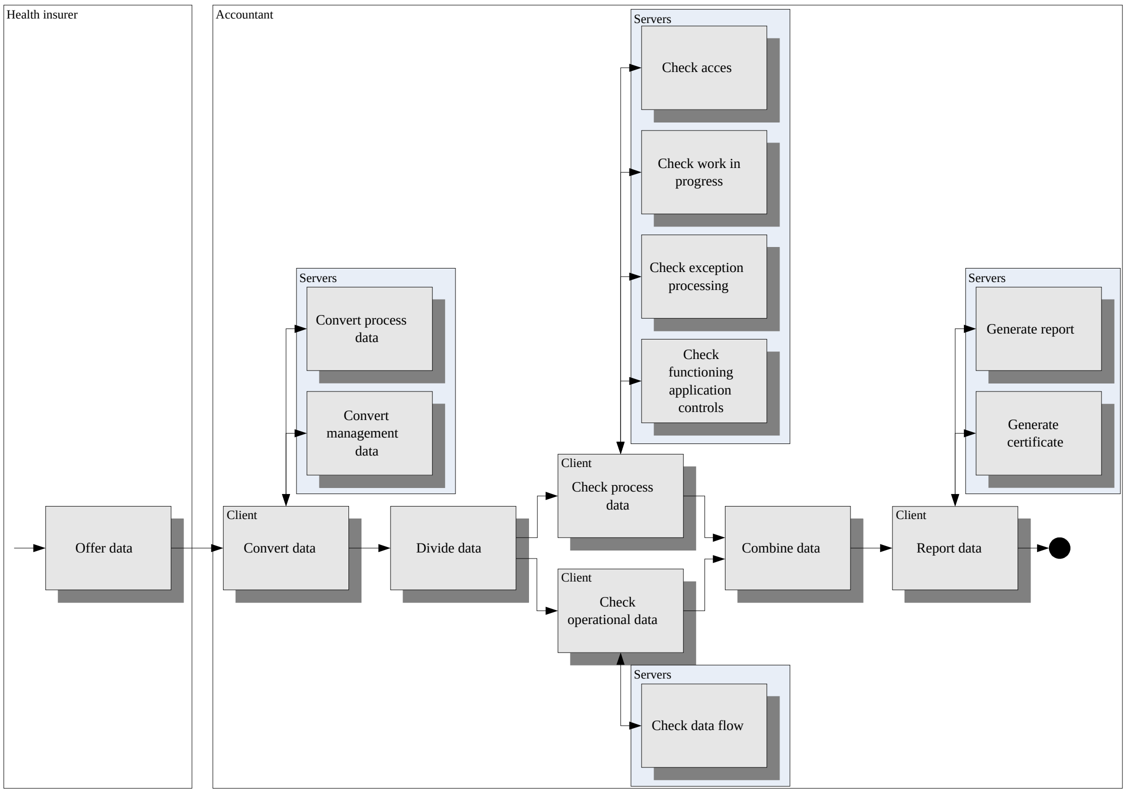

Figure 8.5: Pipe and filter architecture extended with the client-server patterns

Interaction only takes place if exceptions are generated by a filter. Therefore, user interfaces are seen as part of the filter, each filter defines an interface for exception processing.

As already mentioned, management data provides input for the examination done by the functional modules. Figure 8.6 gives an overview of the functional modules using management data encapsulated by the *reflection* pattern. Recall that Figure 6.5, in Section 6.4, shows management data being handled by the client coordinating its services. Each service needs different parts of the management data. If the clients has to record what data each service need, a lot of administration is necessary. An implementation with each service obtaining the right management data from a data resource itself, shown in Figure 8.6, is more convenient and simple.

The architecture in Figure 8.5 does not show which processing steps initiate data to be sent and to be processed. Therefore Figures 8.7 and 8.8 show sequence diagrams, intended to represent the interactions between objects, in this case the filters from the pipe and filter architecture, in the sequential order that interactions occur. More information on sequence diagrams can be found in Appendix B. Figure 8.7 shows operational and process data sent, at regular intervals, to a buffer where data is stored. Figure 8.8 shows data obtained from the buffer and processed by the audit system. Both sequences are modeled separately to emphasize the independence of the audit and the receipt of data. The realization of the audit is initiated by the accountant, even though this may change when moving further to continuous

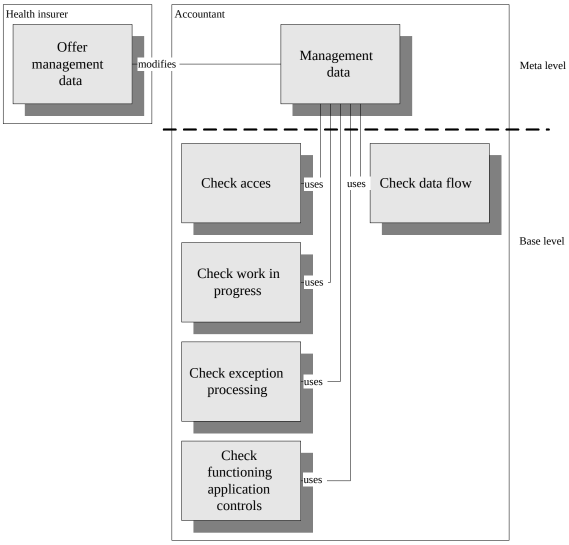

Figure 8.6: Functinal modules using management data

auditing. This is discussed in Section 8.3. Section 8.3 also discusses the actual location of the buffer (at the health insurer or within the audit system) and the way management data is sent and handled.

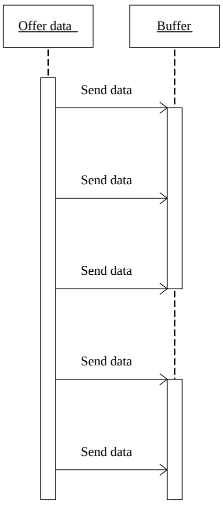

Figure 8.7: Sequence diagram interaction filters health insurer

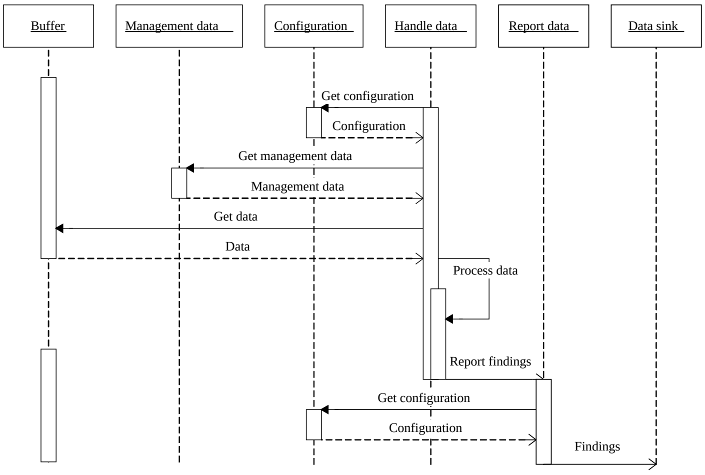

Figure 8.8: Sequence diagram interaction filters accountant

Some attention needs to be paid on data written to and read from the buffer, since it should not be possible that the health insurer and accountant write to and read from the buffer simultaneously, because then there is a chance that the accountant uses incomplete data. A way to overcome this problem, is through the use of a transaction manager, a software component that ensures the buffer to remain in a consistent state despite system failures, and that ensures concurrent transactions to be executed without conflicting [SKS02].

Recommended technology

As for the implementation of the filters, we recommend the use of standard technology. Below, a recommendation is given of what technology to use for each filter in the designed architecture. For each subject, a reference to available literature is provided.

Convert and divide data ETL stands for extraction, transformation, and loading and is used in the context of data warehousing for loading data correctly into the warehouse. Not only is ETL intended for data warehousing, ETL can be used to load data into any database. A variety of commercial ETL tools exist, and also a lot of research effort. Adzic et al. [AFS07] give an overview and Gartner Research [Gar07] provides a recent market review of data integration tools of which ETL forms a part. Kimball et al. [KRT^{+}98] provide in “The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses” an extensive description on how to extract, transform and load data into the proper database and the right format.

Check process and operational data, and combine data Rule-based systems represent knowledge in terms of rules to define conclusions in different situations. Figure 8.9 shows a block diagram of a rule-based system. The knowledge base contains rules representing domain-specific knowledge. The working memory contains the problem-specific facts and conclusions derived by the inference engine. The inference engine uses the information in the working memory together with the rules in the knowledge base to derive conclusions [Dur02].

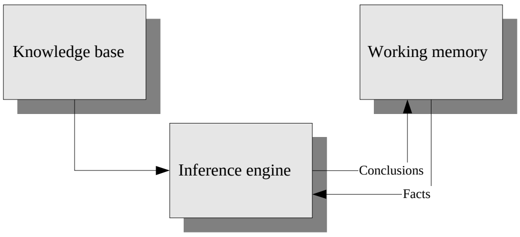

Figure 8.9: Rule-based model [Dur02]

There are two ways a rule-based system operates: backward and forward chaining. Backward chaining starts with a goal to be proved, the system tries to prove the goal by using the rules and checking the facts. Intermediate conclusions that help to support the proof, are also stored. Forward chaining attempts to derive as much information as possible from available facts. The facts act as at trigger for firing rules that add new information to the working memory, which again causes new rules to fire. This process continues until no rule can fire anymore. Afterwards, the working memory contains both the initial facts and all the information inferred by the system [Dur02]. In the audit system, backward chaining can be used for proving that the assertions hold, forward chaining can be used to discover (financial) consequences of invalid assertions. Rule-based systems offer the ability to perform inexact reasoning: they can process uncertain and ambiguous information, and inexact domain knowledge. Inexact reasoning is managed by using certainty factors. More on this subject can be found in [DD98].

Section 3.3 on the scope of this research, states that efficiency and effectiveness of operations can also be of concern for an audit. In the light of efficiency and effectiveness, process mining techniques can be used with process data, to mine the real executed process and analyze it. Process mining techniques allow extracting information from event log files. From this information, formal models of the related business processes can be generated. The information can also be used for comparing observed events with predefined models or business rules. Throughput times and waiting are examples of performance measures on mined processes. Tools for process mining are currently developed, an example is ProM. More information can be found in [DvdAtH05].

Report data Most programming languages and query languages offer standard reporting functionalities. XML, a widely known and applied standard, is used as a format for the reports. XML stands for eXtensible Markup Language, it is a universal standard, designed to present data independent of the application and it is used to exchange structured data. *Markup* refers to information in a document not meant to be displayed. A *markup language* defines which parts of a document are content, which parts are markup and what the markup means.

For automatically processing XML documents, the structure has to be predefined. This can be achieved by a schema or a DTD (Document Type Definition). For the layout of an XML document, XSL (eXtensible Stylesheet Language) is commonly used. XSL prescribes how to display different units of data or elements (identified by so called *tags*) of an XML document. To convert an XML document into another document, XSLT (XSL Transformation) can be used. In our case, this is necessary if the health insurer or government requires a different document than the output of the audit system.

More information on XML can is provided by the World Wide Web Consortium, the owner of the XML standard [Wor].

Following the requirements from Section 5.3, the certificate should be trustworthy, it should be hard to forge the certificate. To this end, electronic certificates are used. An electronic certificate is an electronic document that verifies the sender or receiver's identity. Different kinds of information can be stored in a certificate, including name, serial number, a digital signature, and the expiration date for the certificate. The identity of the persons or organizations that needs to be trusted, is guaranteed by a trusted third-party organization, called a certificate authority. The authority issues certifying authority certificates as a proof of trust [SB02]. In the future, the NIVRA (Dutch association for certified public accountants) could serve as a certificate authority. To guarantee the identity of the NIVRA, the IFAC (International Federation for Accountants) could serve as a certificate authority and grant a certifying authority certificate to the NIVRA.

8.2.2 Middleware layer

This subsection recommends technologies for the middleware layer of the audit system. Middleware is the set of services, protocols, and support utilities providing the 'plumbing' that makes modern distributed systems and applications possible—the infrastructure that underlies web services, distributed systems, e-commerce systems, etc. [SSRB00] For the audit system, the middleware provides a means for connecting the filters, using the components and storing the data. In the application layer, the use of services is already introduced, this is supported by service-oriented architectures.

These days, service-oriented architecture (SOA) is a popular topic. SOA is defined by Tosic et al. [TMP01] as:

> "A software architecture where monolithic applications are decomposed into distributed network-accessible service components, potentially provided by different business entities."

The proposed client-server architecture of the audit system, given in Figure 8.5, introduces service components, therefore a SOA fills our needs. A SOA implementation provides the flexibility we want to achieve by interoperating service components and the possibility to add service components dynamically. Besides, the possibility to make the audit system distributed can be interested with a view to the necessary computational power for data checking.

For the implementation of a SOA, different technologies can be used. We discuss SOAP and ESB to provide a view on the possibilities of SOA technologies, but other technologies are also possible. Simple Object Access Protocol (SOAP) provides a simple and lightweight mechanism for exchanging structured and typed information between modules in a decentralized, distributed environment using XML. SOAP defines a mechanism to pass commands and parameters between HTTP clients and servers. The most commonly used messaging pattern in SOAP is the Remote Procedure Call, based on the *client-server* pattern and therefore suitable for the audit system subject of this research. More on SOAP can be found in [STK02]

A different technology for implementing a SOA, is the Enterprise Service Bus (ESB), which is an event-driven and standards-based messaging engine. The ESB provides an abstraction layer on top of an implementation of an enterprise messaging system, to provide messaging without writing code. The ESB serves as a message broker between applications. The ESB uses XML as the standard communication language. The biggest difference with SOAP is, that with SOAP applications are connected as peers and with ESB applications are connected via the bus. More on ESB can be found in [Cha04].

The rule-based engine and the report generator, discussed in the previous subsection, are also part of the middleware. They do not contain domain-specific business logic. However, the domain-specific rules offered to the rule-based engine contain business logic and are therefore part of the application layer. The configuration and management data are stored in relational databases. The buffer stores operational and process data from the health insurer, a relational database can also be used as buffer, especially if data is stored for a long time, and specific data from the buffer is necessary to perform the audit on. Relational databases conform to relational models, using a collection of tables to represent both data and relationships among those data. More on relational databases can be found in [SKS02].

8.3 Communication audit system and premium process

An important issue concerning the audit system, is the communication with the health insurer. The communication has to be secure and the health insurer should adapt its internal systems as little as possible for the purpose of the communication, following from the requirements in Section 5.3. A natural choice for a communication medium is the Internet. The Internet is cheap to use, provides fast data transfer, and is widely available to everyone. In relation to using tapes, the Internet is cheaper and easier to use. Concerning the band with, taking into account the high amount of transaction of a health insurer every day, the Internet fulfills the need: many gigabytes can be sent every day (compare this with downloading movies). People feel insecure about security of the Internet, though secure protocols for sending data are already available. SSL is widely used on the Internet, for secure communication and encryption of data [SB02].

As already mentioned in Section 7.2 on the organization viewpoint, the health insurer is responsible for making data available, most likely on a server granting access to the audit system. It is the health insurer's responsibility to take security precautions, e.g. installing a fire wall. The accountant is responsible for collecting the data and keeping the data safe and private. The communication can be modeled as a *client-server* architectural pattern, with the audit system as the server, requesting data from the health insurer. In the light of SOA, mentioned in Subsection 8.2.2, the most convenient way to send data to the audit system is by using XML. Furthermore, the fact that XML is widely accepted, makes this a convenient choice.

In Section 6.4, the difference between collecting data real-time or deferred is mentioned. At this moment, there is no need for collecting operational data or process data real-time. The audit concerns a historical period and the audit system takes processing time, requesting an opinion every second or minute would not be interesting, since the opinion is already outdated when received. In the future, when moving further towards continuous auditing, mentioned in Section 2.3, it can be desirable to collect data real-time. Even if the audit is not continuously performed, but only once in a short period, it can be practical to collect data real-time: when the audit system starts performing the audit, data is already collected. Then the setup time for the audit decreases, compared to obtaining data only after the audit process is initiated. Real-time data collection requires the health insurer to have all data up-to-date. If transactions are processed with a substantial delay, real-time data collection is not useful. Currently, in the designed audit system, the accountant is responsible for obtaining data and initializing the audit process. With continuous auditing, the responsibility moves towards the health insurer: the health insurer sends data and initializes the audit process. This can only be achieved, if systems of the health insurer are adapted to this functionality, which implies more modifications of the health insurer's systems.

Management data differs from operational data and process data, since it remains static. Management data changes only if decisions are made on tactical level. Real-time management data collection is practical in this case, since only changes need to be communicated when they take place.

In the application layer, we introduced a buffer for storing data that needs to be processed. For storing this data, two options are available: storing data on the health insurer's side, or on the accountant's side. Storing data at the accountant's side is recommended, because data can already be converted and the setup time for the actual audit process becomes less. Besides, when an audit is started, the accountant is not dependent of the health insurer for obtaining the correct data, since the data is already stored at the accountant's side.

Chapter 9

Conclusion

This research provides an architectural framework for IT-enabled, continuous auditing of the premium process at health insurers. During the research we realized that providing an architectural framework for IT-enabled, continuous auditing of all processes at health insurers would be too complex. Therefore, we decided to focus on the premium process. During this research, a sequence of activities has taken place: the premium process is analyzed, the requirements for an audit system supporting IT-enabled, continuous auditing are elicited, and a process viewpoint, a data viewpoint, an organization viewpoint, and a system viewpoint of the audit system are developed. The aim of this research is to investigate the possibilities of IT-enabled, continuous auditing, i.e. is IT-enabled, continuous auditing feasible? This question is answered in Section 9.1. Section 9.2 gives an overview of subjects that need to be investigated further, before an implementation of an IT-enabled, continuous audit system would be possible. A change management plan with recommended steps to come to an implementation of the audit system is presented in Section 9.3. Section 9.4 discusses contributions of this research to PwC and the academic world. In Section 9.5, a reflection of the developed product, which is the architectural framework, and the followed process approach is given.

9.1 "Is IT-enabled, continuous auditing feasible?"

Technically speaking, for auditing the premium process, the answer is yes. This thesis provides a design for a system supporting IT-enabled, continuous auditing of the premium process at health insurers. All technologies recommended in Chapter 8 for a future implementation, are currently available. Though some remarks are made on the risks expected with an implementation. First, filling the functional modules, explained in Section 8.2.1, with domain-specific rules could be very complex. Second, for the electronic certification of the accountant and its audit system, the accountant is dependent of trusted third parties, willing to issue electronic certificates. Third, the accountant is dependent of the cooperation of its client, in this case the health insurer. If the client does not want to cooperate by providing the requested data automatically and digitally, the audit system is not useful. An additional risk can be expected within the accountancy firm. Among employees, resistance could be experienced when developing the audit system. This internal organizational problem is discussed in Section 7.2, on the organization viewpoint.

Is the audit system extendible to different processes at the health insurer, or even to processes at other organizations? For the health insurer, the answer is yes. Most processes of the health insurer are automated, and overviews of different processes similar to the functional description of the premium process can be made. For manual processes, not following strict procedures and having standardized output, IT-enabled, continuous auditing is not advisable. For other organizations, it depends on the nature of the organization. In Mintzberg's vision on organizational structures, a health insurer is classified as a machine bureaucracy [Min83]. The machine bureaucracy is characterized by simple routine operations and standardized work processes. As a result, processes of machine bureaucracies are suitable for automation, a precondition for applying the audit system. Mintzberg also states that the external control, including the external audit, is high in many machine bureaucracies. IT-enabled, continuous auditing is therefore desirable, especially since health insurers are also typified by a high amount of transactions every day.

The question "Is *continuous* auditing feasible?" has not sufficiently been answered yet. The answer depends on the frequency of the audit and the topicality of the data to be examined. An audit provides an opinion for a certain period. From this view, performing an audit every second is not useful. Besides, performing the audit also takes time, so when the output is available, it is already outdated. To perform an audit weekly or monthly on last week's or last month's data is feasible, provided that the data to be investigated is topical. This is often the case with highly automated organizations.

A last remark is made on the positioning of the audit within the three levels, strategic, tactical, and operational, of an organization. In Chapter 3, Figure 3.1 shows that the audit is performed at the strategic and the tactical level. When performing audits continuously using the audit system, it becomes an operational process and we see the operational audit process within the audit system shift to the operational level, see Figure 9.1. The correct functioning of the audit system depends on the configuration of the audit system, e.g. providing the system with the correct domain-specific rules. At the tactical level of the audit, what we call "audit control", the results of the audit are communicated to the management, and decisions on the system configuration are made.

9.2 Recommendations

In the previous section, four risks have been indicated concerning a future implementation of the audit system: the complexity of domain-specific rules, the electronic certification of the accountant and audit system, the cooperation of the client, and the acceptance within the accountancy firm (see Section 7.2). Further research should reveal the complexity of providing the system with correct domain-specific rules. For the electronic certification, the accountant depends of the cooperation of a trusted third party. In our opinion, the NIVRA (Dutch association for certified public accountants) would qualify for this. Explorative interviews

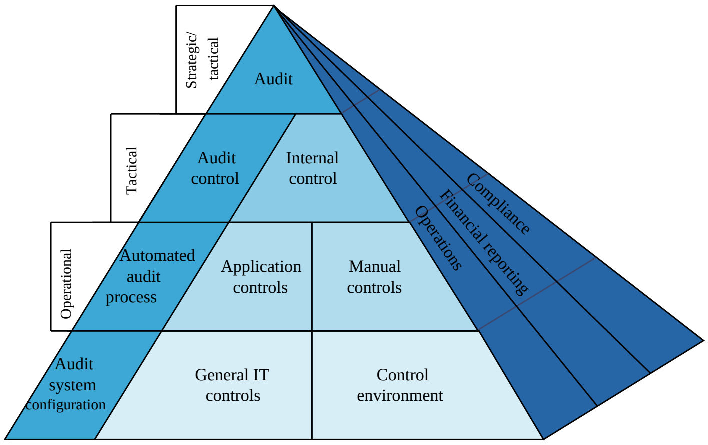

Figure 9.1: Shift of audit in framework of audit dependencies and scope

with the NIVRA should tell the practicability of the NIVRA as a certificate authority. As for the collaboration with the client, the possibilities for strategic alliances should be further explored.

Before the designed audit system can be implemented, the architectural framework provided by this research should be elaborated in depth. The processes need to be specified further, all data used and produced by the audit system should be mapped out, specific responsibilities need to be assigned and specific technologies should be chosen for an implementation. A market review of available tooling is recommended, to use with a future implementation. Also the requirements in Chapter 5, on which the audit system is based, need to be reviewed formally. The influence of XBRL, discussed in Section 2.4.4, for the audit system could also be further investigated. When XBRL becomes a standard for business reporting and financial reporting, and rules concerning financial reporting are standardized and digitally available, the audit system can be extended to perform checks on the financial reporting and the financial statements. Checks on reporting and on the statements can be added as functional modules to the audit system. The next section recommends a plan for further research and an implementation of the audit system.

9.3 Change management plan

This section provides a plan for change management for a first implementation of the audit system in the future. This plan is included, since effective management of changes is a key success-factor for an enterprise [Col01]. The goal of change management is to provide a disciplined process for introducing required changes into the environment with minimal disruption to ongoing operations. It is important that the people involved in changing processes are able to understand, accept and support the changes, therefore the following activities are fundamental: information, communication, and education [Ham95].

The previous section mentions work to be done before starting with an implementation of the audit system. This includes research into domain-specific rules, explorative interviews with the NIVRA or another organization qualified for issuing certifications, and investigating possibilities for collaboration with a client. Also the opportunities for introducing changes step-by-step within PwC have to be investigated. This results in the following parties to be involved: A client, possibly a health insurer but another client with the same characteristics can be chosen as well, the NIVRA or a similar organization, and the accountants, the management and IT-specialists within PwC.

Figure 9.2 gives a roadmap with the phases of a change management project for a first implementation of the audit system. First a client needs to be found willing to cooperate. Business rules for the domain of the client need to be gathered and analyzed to gain insights in the complexity and, as a result, feasibility of implementing the system. Parallel to this step, the possibilities for certification by a certificate authority should be investigated. If both steps achieve satisfactory results, and an implementation seems feasible, a collaboration between the client and PwC should be established, e.g. in the form of a strategic alliance.

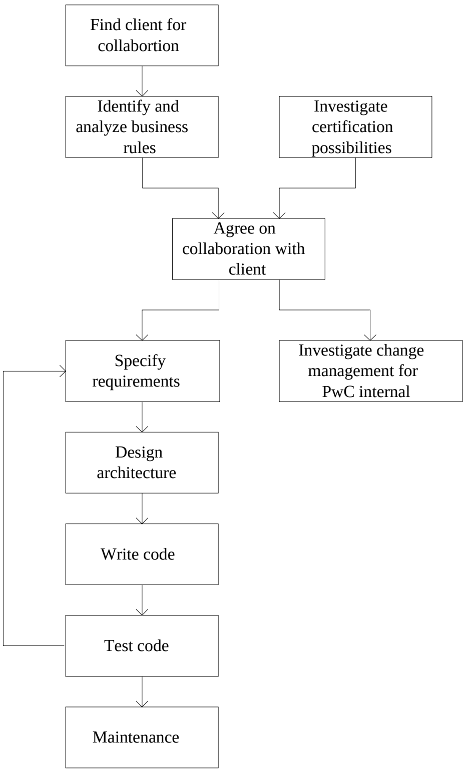
Find client for collabortion
Identify and analyze business rules
Investigate certification possibilities
Agree on collaboration with client
Specify requirements
Investigate change management for PwC internal
Design architecture
Write code
Test code
Maintenance

Figure 9.2: Roadmap for implementing the audit system

The next phases are focussed on the technical implementation. The requirements have to be extended and managed, see Chapter 5, and the functional design and the four viewpoints in Chapters 6, 7, and 8 need to be extended. Subsequently the system can be implemented and tested, and a plan for maintenance should be provided. For software engineering projects, a variety of approaches is available. From personal experience, we suggest using an iterative way of software engineering, e.g. eXtreme Programming [BA04]. This approach can handle the change of requirements during a project and starts with a small design and implementation, improving and extending it incrementally. As a result, all necessary functionality is build first, and a working system is promptly available so that stakeholders can already see the added-value.

During the technical implementation, accountants should be highly involved to create support for the audit system. Requirements and decisions should be based on the accountant's input. During the technical implementation, research can be done into the possibilities for introducing changes at PwC, by means of e.g. workshops, changes in the education, and providing up-to-date information on the progress of the implementation.

In the future, if rules for financial reporting are more standardized, and XBRL is widely used, the audit system can be extended to check financial reporting as well, as mentioned in Section 9.2.

This section provides a change management plan for a first implementation. Subsequent implementations for different clients are less time consuming. The steps on the investigation of certification possibilities, change management for PwC internal, and the design and implementation of the system are not necessary since an implementation is already available. Because of the flexibility of the system, the system can be configured to different customers. Identifying and analyzing business rules, and maintaining the audit system remain important activities.

9.4 Contributions

For PwC, this research provides insights into the feasibility of IT-enabled, continuous auditing. It provides a practical concept of an application of IT-enabled, continuous auditing, which makes the subject more tangible for accountants. This research provides a starting point for PwC to investigate IT-enabled, continuous auditing further. Besides being an accountancy firm, PwC operates also as a consultancy firm. Therefore, this research can be used to form a vision on this subject, which can be communicated to clients interested in IT-enabled, continuous auditing. Also for the internal strategy of PwC, from which the desire to move further towards continuous assurance, mentioned in Chapter 2, might speak in the future, this thesis provides a basis.

As for the scientific contribution of this research, the architectural framework we developed has characteristics of a reference architecture. A reference architecture is defined in "Software Architecture in Practice" as a reference model (a division of functionality with its data flows) mapped onto software elements, and the data flows between them [BCK03]. It is often used for a particular domain. The framework in this thesis can be used as a reference for future implementations of IT-enabled (continuous) auditing, for each industry.

In Section 2.4, different technologies to provide continuous assurance are discussed. A difference between the audit system and the use of mobile agents is that no additional software has to be installed at the client. Clients are not in favour of running software on their systems, which is managed by a third party. For using web services, all processes need to be defined formally. This is too costly for clients, especially when other, less time consuming, methods are at hand. The functionality used with data marts is based on statistical analysis, individual facts are not checked. In the audit system, all transactions can be checked. Data marts can be useful if it is not known in advance when an audit needs to be performed. An analysis of historical data can be provided any time. In contrast, the audit system is based on an ongoing data stream to be checked regularly, and is therefore a continuous process.

For developing the architectural framework, we used IT knowledge to solve accountancy related problems. Applying IT solutions to the accountancy domain contributes to the current state of the art in both domains.

9.5 Reflection

This section reflects on the final product, which is the architectural framework, and the process approach of this research.

Architectural framework The architectural framework of IT-enabled, continuous auditing, presented in this thesis, provides a concrete concept for the application of IT-enabled, continuous auditing. The level of detail is appropriate for getting an impression of its utility and feasibility, which is the aim of this research. To implement the audit system, the design should be elaborated into more detail. More decisions on the design have to be made, which implies more discussions with the users of the audit system. To create support for the system, more stakeholders should be involved. Discussions and more involved stakeholders result in the need of a bigger effort and more time to design the audit system.

The decision to develop an architectural framework with a process viewpoint, a data viewpoint, an organization viewpoint, and a system viewpoint, has been convenient. These viewpoints direct the functionality and the organization of an architecture, more than the implementation, which is suitable for our research. Other viewpoints of different frameworks can be filled in when elaborating the design, e.g. an application viewpoint or an infrastructure viewpoint, which are basically extensions of the viewpoints amplified in this research. Choosing a single framework, e.g. TOGAF or the Zachman Framework [LTP$^+$05], and selecting viewpoints from this framework is recommended, in order to develop a complete and consistent design.

The architectural framework has proven to be functional in the communication with the stakeholders, especially through the use of standard, widely used modeling techniques and through the effort of keeping designs simple. Providing graphical overviews in a standardized way reveals incorrect communication and incomplete overviews. Keeping different levels of detail within a model, improved the understanding: if a level is too complex due to its details, a higher level overview with less details gives insights in the structure and makes it easier to explain a lower level overview.

Process approach The process approach of this project has not been the most appropriate choice. Too much time was taken for analyzing the processes at health insurers. It would have been better, if we had chosen an approach which explicitly indicated when domain-specific knowledge was necessary. This would have shifted the emphasis on analyzing all processes within a health insurer to only the information of health insurers necessary for this research. Besides, focussing on the premium process could have been decided earlier by using an approach distinguishing domain-specific knowledge. On the other hand, knowing so much about processes at health insurers made it easier to give examples of the specific use of the audit system and its modeled functionality. This made communication with the stakeholders easier. To summarize, the followed approach has proven to be effective, but not efficient. Nevertheless, mapping out all processes provided experience in modeling 'real life' processes, and therefore has been very valuable.

Tackling specific accountancy knowledge has also been a great effort; a more structured approach for this would have been useful, though it is hard to use an appropriate approach without any domain knowledge. This is probably inherent in bridging different, unknown domains. Looking back, a great challenge of this graduation project was the mapping of the user domain on the IT domain: gaining knowledge from accountants and employees at a health insurer, translating this into the IT and business domain, developing new IT solutions, and communicating them back to the accountants and employees in a way they understand.

The order in which the viewpoints were designed (process, data, organization, system) has proven to be functional. The first question was what the system should do. A clear view had to be developed before answering the question on how this should be achieved. Therefore the functional description and related process viewpoint were profitable. The data and organization viewpoint are based on the functional description, changing the order of these viewpoints would not have been useful. The data and organization viewpoint are designed independently; designing them could have been interchanged. The system viewpoint recommends technology for reporting and certification, enforced by decisions made for the data viewpoint. Also the communication part of the system viewpoint takes assigned responsibilities in the organization viewpoint into consideration. Therefore, the system viewpoint should be the last viewpoint to design.

Within the functional description, the organization levels (strategic, tactical, and operational) are filled in bottom up, because the operational processes of the audit system, containing its main functionality, were most important. Without having the operational level defined, a definition of the tactical and strategic level would have been difficult to give. Within each level, a top down approach is used. This helped in the communication to the stakeholders, as already discussed above, with the reflection of the architectural framework. A bottom up approach would have caused us to get lost in details.

Now a conceptual design for IT-enabled, continuous auditing is provided, the way to continuous assurance seems a little shorter. Supporting technologies are widely available, so further developments of audit-related IT possibilities are not hindered by technical limitations anymore. Accountants should no longer deny the opportunities of IT within their domain, but a new world should be created where accountancy and IT go hand in hand.

Bibliography

[ABB01] J.E. van Aken, J.D. van der Bij, and J.J. Berends. Bedrijfskundige methodologie, collegedictaat. Technical report, Technische Universiteit Eindhoven, 2001.

[AFS07] Jovanka Adzic, Valter Fiore, and Luisella Sisto. Extraction, transformation, and loading processes. In Robert Wrembel and Christian Koncilia, editors, *Data Warehouses and OLAP: Concepts, Architectures and Solutions*, pages 88–111. IRM Press, 2007.

[AIC07] AICPA (The American Institute of Certified Public Accountants). Statements on auditing standards no. 106: Audit evidence, 2007. `http://www.aicpa.org/download/members/div/auditstd/AU-00326.PDF`.

[Akt87] A. Z. Aktas. *Structured Analysis and Design of Information Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1987.

[AZ05] P. Avgeriou and U. Zdun. Architectural patterns revisited - a pattern language. In *Proceedings of 10th European Conference on Pattern Languages of Programs (EuroPlop 2005)*, Irsee, Germany, July 2005.

[BA04] Kent Beck and Cynthia Andres. *Extreme Programming Explained: Embrace Change (2nd Edition).* Addison-Wesley Professional, 2004.

[BCK03] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice, Second Edition.* Addison-Wesley Professional, April 2003.

[BMR$^+$96] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, and Michael Stal. *Pattern-Oriented Software Architecture — A System of Patterns.* Wiley & Sons, New York, NY, USA, 1996.

[CCA06] David R. Campbell, Mary Campbell, and Gary W. Adams. Adding significant value with internal controls. *The CPA Journal*, 76(6):20–25, June 2006.

[Cha04] David Chappell. *Enterprise Service Bus.* O'Reilly, July 2004.

[CIC99] CICA/AICPA. Continuous auditing, research report. Technical report, CICA/AICPA, Toronto, Canada, 1999.

[Cod05] David Coderre. Gtag – continuous auditing: Implications for assuranc, monitoring and risk assessment. Technical report, Institute of Internal Auditors (IIA), 2005.

[Col01] James Collins. *Good to Great.* HarperBusiness, New York, 2001.

[COS04] COSO (The Committee of Sponsoring Organizations of the Treadway Commission). *Enterprise Risk Management — Integrated Framework.* AICPA, 2004.

[DD98] Jack Durkin and John Durkin. *Expert Systems: Design and Development.* Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[Dur02] John Durkin. *Tools and Applications*, volume 1 of *Expert Systems: The Technology of Knowledge Management and Decision Making for the 21st Century*, pages 23–52. Academic Press, 2002.

[DvdAtH05] M. Dumas, W. M. van der Aalst, and A. H. ter Hofstede. *Process-aware information systems: bridging people and software through process technology.* John Wiley & Sons, Inc., New York, NY, USA, 2005.

[Gar07] Gartner Research. Magic quadrant for data integration tools, 2007. `http://mediaproducts.gartner.com/reprints/businessobjects/151150.html`.

[GF99] Marilyn Greenstein and Todd M. Feinman. *Electronic Commerce: Security Risk Management and Control.* McGraw-Hill Higher Education, 1999.

[Ham95] Michael Hammer. *The Reengineering Revolution.* HarperBusiness, New York, 1995.

[IEE00] IEEE Architecture Working Group. IEEE std 1471-2000, recommended practice for architectural description of software-intensive systems. Technical report, IEEE, 2000.

[KK00] P.M. Kempen and J.A. Keizer. *Advieskunde voor praktijkstages: Organisatieverandering als leerproces.* Wolters-Noordhoff B.V., 2000. In Dutch.

[KRT+98] Ralph Kimball, Laura Reeves, Warren Thornthwaite, Margy Ross, and Warren Thornwaite. *The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing and Deploying Data Warehouses with CD Rom.* John Wiley & Sons, Inc., New York, NY, USA, 1998.

[Kru95] Philippe Kruchten. Architectural blueprints—the “4+1” view model of software architecture. *IEEE Software*, 12(6):42–50, November 1995.

[LTP+05] M.M. Lankhorst, L. van der Torre, H.A. Proper, F. Arbab, and M.W.A. Steen. *Enterprise Architecture at Work : Modelling, Communication and Analysis.* Springer, Berlin, Germany, EU, 2005.

[MG04] Uday S. Murthy and S. Michael Groomer. A continuous auditing web services model for xml-based accounting systems. *International Journal of Accounting Information Systems*, 5:139–163, 2004.

[Mil03] Randy Miller. Practical uml?: A hands-on introduction for developers, 2003. `http://dn.codegear.com/article/31863`.

[Min83] H. Mintzberg. *Structure in Fives.* Prentice-Hall, Englewood-Cliffs, New Jersey, 1983.

[NE00] Bashar Nuseibeh and Steve Easterbrook. Requirements engineering: a roadmap. In *ICSE - Future of SE Track*, pages 35–46, 2000.

[Pri04] PricewaterhouseCoopers. PwC Audit 2004 Quick reference guide. Internal publication, 2004.

[RSE02] Z. Rezaee, A. Sharbatoghlie, and R. Elam. Continuous auditing: Building automated auditing capability. *Auditing: A Journal of Practice & Theory*, 21(1):147–163, March 2002.

[SB02] Craig Van Slyke and France Blanger. *E-Business Technologies: Supporting the Net-Enhanced Organization.* Wiley, 2002.

[SEI07] SEI (Software Engineering Institute) – Carnegie Mellon. SEI open systems glossary, September 2007. `http://www.sei.cmu.edu/opensystems/glossary.html#s`.

[SJBA98] Richard Stevens, Ken Jackson, Peter Brook, and Stuart Arnold. *Systems engineering: coping with complexity.* Prentice Hall, London, 1998.

[SKS02] Abraham Silberschatz, Henry F. Korth, and S. Sudershan. *Database System Concepts (fourth edition).* McGraw-Hill, Inc., New York, NY, USA, 2002.

[SMJ98] R.W. Starreveld, H.B. De Mare, and E.J. Joëls. *Bestuurlijke informatieverzorging / 2B Toepassingen.* Wolters-Noordhoff B.V., 1998. In Dutch.

[SSRB00] Douglas Schmidt, Michael Stal, Hans Rohnert, and Frank Buschmann. *Pattern-Oriented Software Architecture, Volume 2, Patterns for Concurrent and Networked Objects.* John Wiley & Sons, September 2000.

[STK02] James Snell, Doug Tidwell, and Pavel Kulchenko. *Programming Web services with SOAP.* O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2002.

[SW04] Graeme Simsion and Graham Witt. *Data Modeling Essentials, Third Edition (The Morgan Kaufmann Series in Data Management Systems).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[TMP01] Vladimir Tosic, David Mennie, and Bernard Pagurek. Software configuration management related to the management of distributed systems and service-oriented architectures. In *SCM*, pages 54–69, 2001.

[TOM$^+$90] J. Truijens, A. Oosterhaven, R. Maes, H. Jägers, and F. van Iersel. *Informatie-infrastructuur, een instrument voor het management.* Kluwer Bedrijfswetenschappen, 1990. In Dutch.

[Wor] World Wide Web Consortium. Extensible markup language (xml). `http://www.w3.org/XML`.

[WS01] Jon Woodroof and DeWayne Searcy. Continuous audit: Model development and implementation within a debt covenant compliance domain. *International Journal of Accounting Information Systems*, 2:169–191, 2001.

[XBR06] XBRL Nederland. Wat is XBRL?, Accessed on September 22, 2006. `http://www.xbrl-nederland.nl`.

[Zav97] Pamela Zave. Classification of research efforts in requirements engineering. *ACM Computing Surveys*, 29(4):315–321, 1997.

List of Figures

Appendices

Appendix A

Data Flow Diagrams

Data Flow Diagrams (DFDs) illustrate how data flows between processes in terms of inputs and outputs. DFDs do not provide information on the order in which data is processed. Furthermore, DFDs do not take the physical environment (e.g. pc, phone) or physical data storage (e.g. email, drives) into account. DFDs use the notations shown in Figure A.1

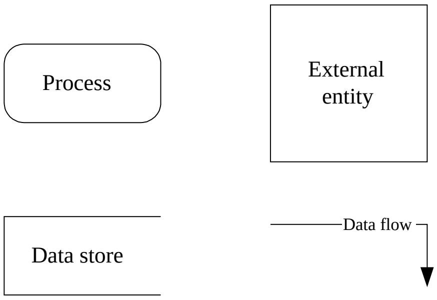

Figure A.1: Data Flow Diagram notation

Process A process transforms incoming data flow into outgoing data flow.

Data store Data stores are repositories of data in the system. They are sometimes also referred to as files.

Data flow Data flows are pipelines through which packets of information flow. The arrows are labeled with the name of the data that moves through it.

External entity External entities are objects outside the system, with which the system communicates. External entities are sources and destinations of the system's inputs and outputs.

DFDs can be designed with nested layers. A single process node on a high level diagram can be expanded to show a more detailed data flow diagram.

More information on Data Flow Diagrams can be found in [Akt87].

Appendix B

Practical UML?: A Hands-On Introduction for Developers

By Randy Miller [Mil03]

The heart of object-oriented problem solving is the construction of a model. The model abstracts the essential details of the underlying problem from its usually complicated real world. Several modeling tools are wrapped under the heading of the UML, which stands for Unified Modeling Language. The purpose of this paper is to present important highlights of the UML. At the center of the UML are its nine kinds of modeling diagrams:

- Use case diagrams
- Class diagrams
- Object diagrams
- Sequence diagrams
- Collaboration diagrams
- Statechart diagrams
- Activity diagrams
- Component diagrams
- Deployment diagrams

A description of use case diagrams, class diagrams, sequence diagrams, and activity diagrams is given below.

B.1 Why is UML important?

Let's look at this question from the point of view of the construction trade. Architects design buildings. Builders use the designs to create buildings. The more complicated the building, the more critical the communication between architect and builder. Blueprints are the standard graphical language that both architects and builders must learn as part of their trade. Writing software is not unlike constructing a building. The more complicated the underlying system, the more critical the communication among everyone involved in creating and deploying the software. In the past decade, the UML has emerged as the software blueprint language for analysts, designers, and programmers alike. It is now part of the software trade. The UML gives everyone from business analyst to designer to programmer a common vocabulary to talk about software design. The UML is applicable to object-oriented problem solving. Anyone interested in learning UML must be familiar with the underlying tenet of object-oriented problem solving – it all begins with the construction of a model. A model is an abstraction of the underlying problem. The domain is the actual world from which the problem comes. Models consist of objects that interact by sending each other messages. Think of an object as "alive." Objects have things they know (attributes) and things they can do (behaviors or operations). The values of an object's attributes determine its state. Classes are the "blueprints" for objects. A class wraps attributes (data) and behaviors (methods or functions) into a single distinct entity. Objects are instances of classes.

B.2 Use case diagrams

Use case diagrams describe what a system does from the standpoint of an external observer. The emphasis is on *what* a system does rather than *how*. Use case diagrams are closely connected to scenarios. A scenario is an example of what happens when someone interacts with the system. Here is a scenario for a medical clinic. "A patient calls the clinic to make an appointment for a yearly checkup. The receptionist finds the nearest empty time slot in the appointment book and schedules the appointment for that time slot. " A use case is a summary of scenarios for a single task or goal. An actor is who or what initiates the events involved in that task. Actors are simply roles that people or objects play. The picture below is a **Make Appointment** use case for the medical clinic. The actor is a **Patient**. The connection between actor and use case is a communication association (or communication for short).

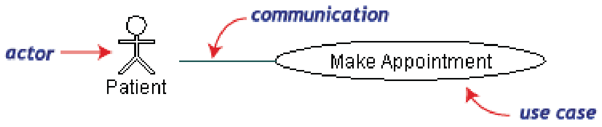

Figure B.1: A use case diagram

Actors are stick figures. Use cases are ovals. Communications are lines that link actors to use cases. A use case diagram is a collection of actors, use cases, and their communications.

We've put **Make Appointment** as part of a diagram with four actors and four use cases. Notice that a single use case can have multiple actors.

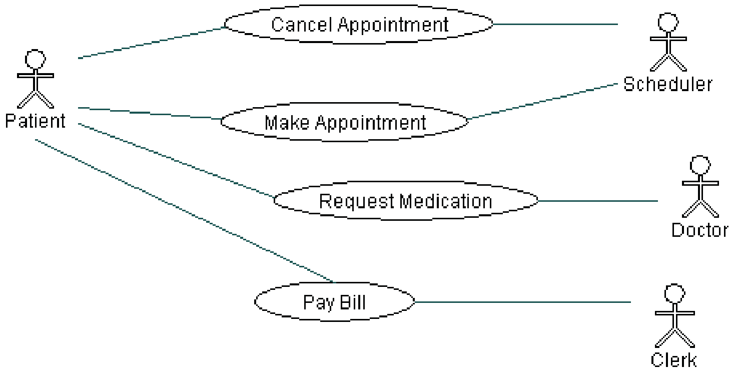

Figure B.2: A detailed use case diagram

Use case diagrams are helpful in three areas.

Determining features (requirements) New use cases often generate new requirements as the system is analyzed and the design takes shape.

Communicating with clients Their notational simplicity makes use case diagrams a good way for developers to communicate with clients.

Generating test cases The collection of scenarios for a use case may suggest a suite of test cases for those scenarios.

B.3 Class diagrams

A Class diagram gives an overview of a system by showing its classes and the relationships among them. Class diagrams are static – they display what interacts but not what happens when they do interact. The class diagram below models a customer order from a retail catalog. The central class is the **Order**. Associated with it are the **Customer** making the purchase and the **Payment**. A **Payment** is one of three kinds: **Cash**, **Check**, or **Credit**. The order contains **OrderDetails** (line items), each with its associated **Item**.

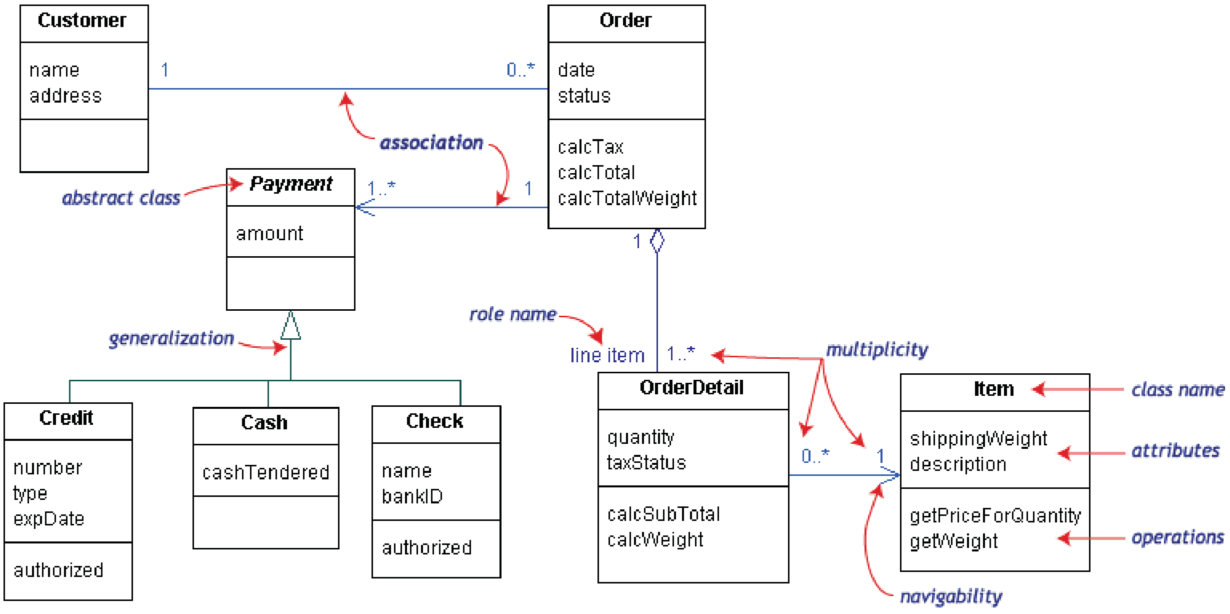

Figure B.3: A class diagram

UML class notation is a rectangle divided into three parts: class name, attributes, and operations. Names of abstract classes, such as ***Payment***, are in italics. Relationships between classes are the connecting links. Our class diagram has three kinds of relationships.

Association A relationship between instances of the two classes. There is an association between two classes if an instance of one class must know about the other in order to perform its work. In a diagram, an association is a link connecting two classes.

Aggregation An association in which one class belongs to a collection. An aggregation has a diamond end pointing to the part containing the whole. In our diagram, **Order** has a collection of **OrderDetails**.

Generalization An inheritance link indicating one class is a superclass of the other. A generalization has a triangle pointing to the superclass. ***Payment*** is a superclass of **Cash**, **Check**, and **Credit**.

An association has two ends. An end may have a role name to clarify the nature of the association. For example, an **OrderDetail** is a line item of each **Order**. A navigability arrow on an association shows which direction the association can be traversed or queried. An **OrderDetail** can be queried about its **Item**, but not the other way around. The arrow also lets you know who "owns" the association's implementation; in this case, **OrderDetail** has an **Item**. Associations with no navigability arrows are bi-directional. The multiplicity of an association end is the number of possible instances of the class associated with a single instance of the other end. Multiplicities are single numbers or ranges of numbers. In our example, there can be only one **Customer** for each **Order**, but a **Customer** can have any number of **Orders**. This table gives the most common multiplicities.

Multiplicities	Meaning
0..1	zero or one instance. The notation n . . m indicates n to m instances.
0..* or *	no limit on the number of instances (including none).
1	exactly one instance.
1..*	at least one instance.

Every class diagram has classes, associations, and multiplicities. Navigability and roles are optional items placed in a diagram to provide clarity.

B.4 Sequence diagrams

Class and object diagrams are static model views. Interaction diagrams are dynamic. They describe how objects collaborate. A sequence diagram is an interaction diagram that details how operations are carried out – what messages are sent and when. Sequence diagrams are organized according to time. The time progresses as you go down the page. The objects involved in the operation are listed from left to right according to when they take part in the message sequence. Below is a sequence diagram for making a hotel reservation. The object initiating the sequence of messages is a **Reservation window**.

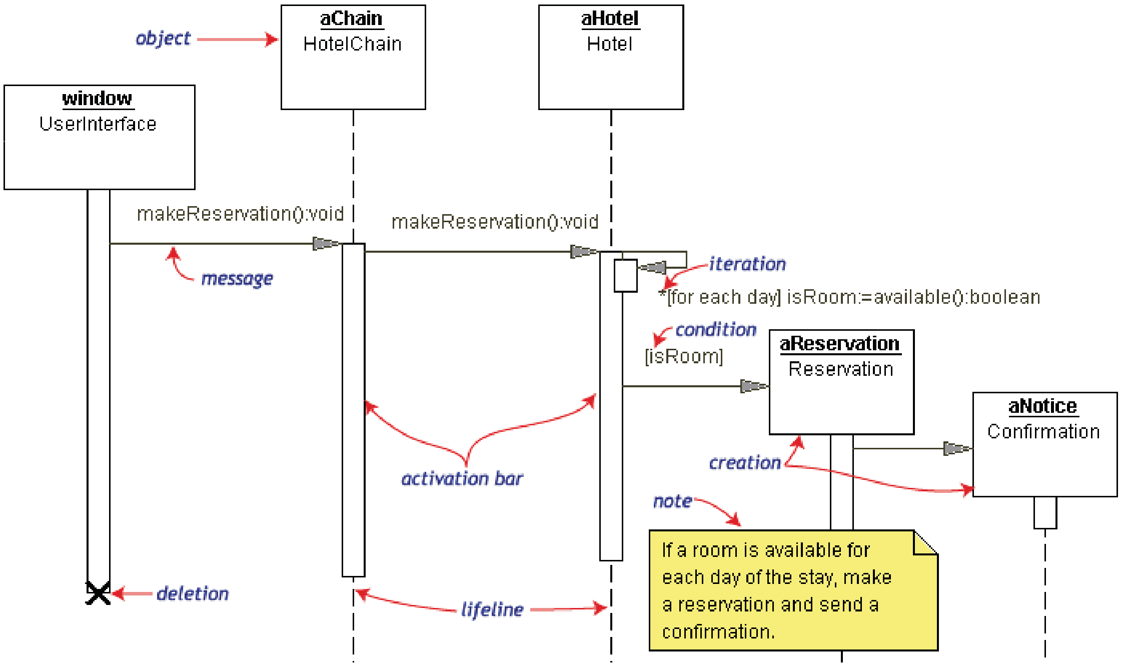

Figure B.4: A sequence diagram

The **Reservation window** sends a `makeReservation()` message to a **HotelChain**. The **HotelChain** then sends a `makeReservation()` message to a **Hotel**. If the **Hotel** has available rooms, then it makes a **Reservation** and a **Confirmation**. Each vertical dotted line is a lifeline, representing the time that an object exists. Each arrow is a message call. An arrow goes from the sender to the top of the activation bar of the message on the receiver's lifeline. The activation bar represents the duration of execution of the message. In our diagram, the **Hotel** issues a self call to determine if a room is available. If so, then the **Hotel** creates a **Reservation** and a **Confirmation**. The asterisk on the self call means iteration (to make sure there is available room for each day of the stay in the hotel). The expression in square brackets, [ ], is a condition. The diagram has a clarifying note, which is text inside a dog-eared rectangle. Notes can be put into any kind of UML diagram.

B.5 Activity diagrams

An activity diagram is essentially a fancy flowchart. Activity diagrams and statechart diagrams are related. While a statechart diagram focuses attention on an object undergoing a process (or on a process as an object), an activity diagram focuses on the flow of activities involved in a single process. The activity diagram shows the how those activities depend on one another. For our example, we used the following process. "Withdraw money from a bank account through an ATM." The three involved classes (people, etc.) of the activity are **Customer**, **ATM**, and **Bank**. The process begins at the black start circle at the top and ends at the concentric white/black stop circles at the bottom. The activities are rounded rectangles.

Activity diagrams can be divided into object swimlanes that determine which object is responsible for which activity. A single transition comes out of each activity, connecting it to the next activity. A transition may branch into two or more mutually exclusive transitions. Guard expressions (inside [ ]) label the transitions coming out of a branch. A branch and its subsequent merge marking the end of the branch appear in the diagram as hollow diamonds. A transition may fork into two or more parallel activities. The fork and the subsequent join of the threads coming out of the fork appear in the diagram as solid bars.

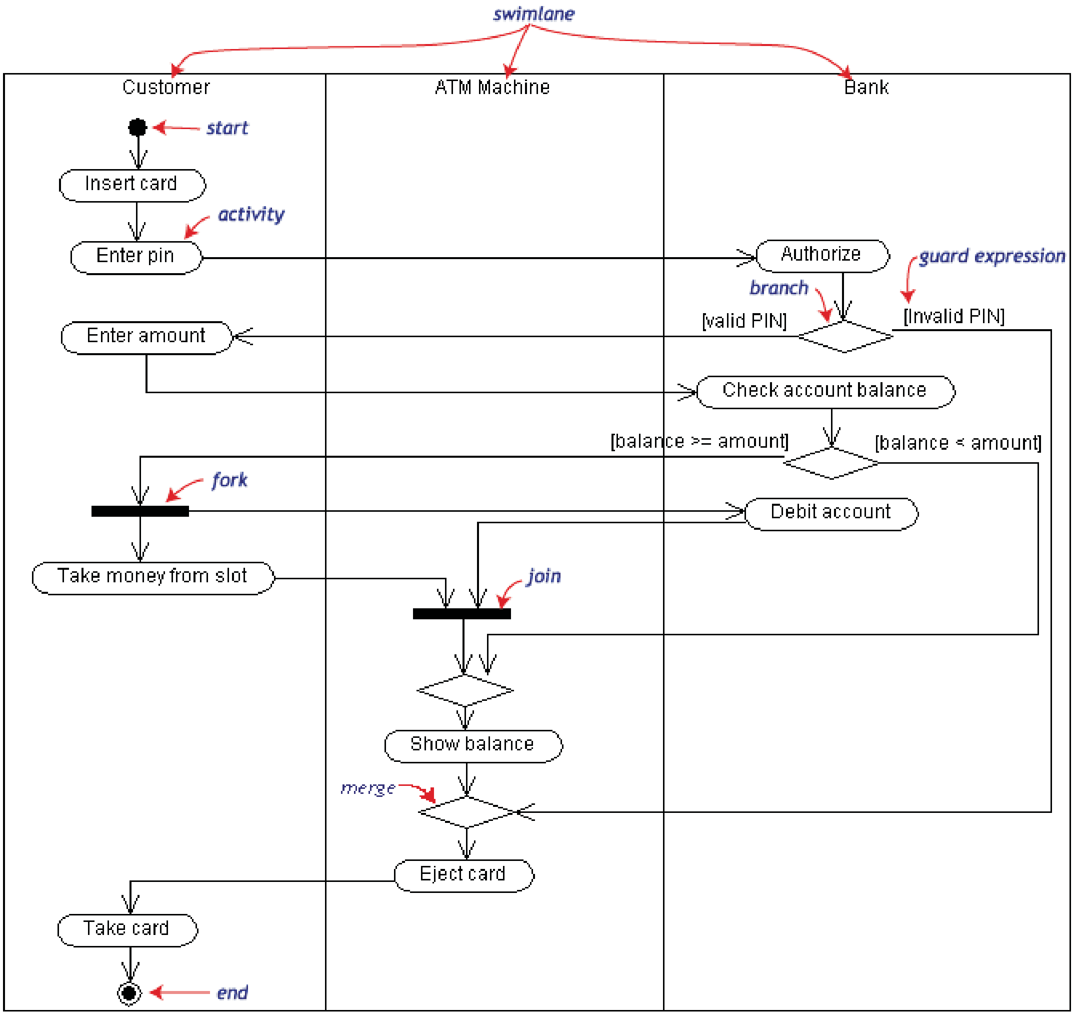

Figure B.5: An activity diagram

Appendix C

Process viewpoint audit system

This appendix provide the process viewpoint of the audit system, related to the functional description in Chapter 6. UML activity diagrams are used to model the process viewpoint, and are explained in Appendix B. Figure C.1 provides the activity diagram for the activities of the audit system at the three different organizational levels: strategic, tactical, and operational. An explanation is given in Section 6.1.

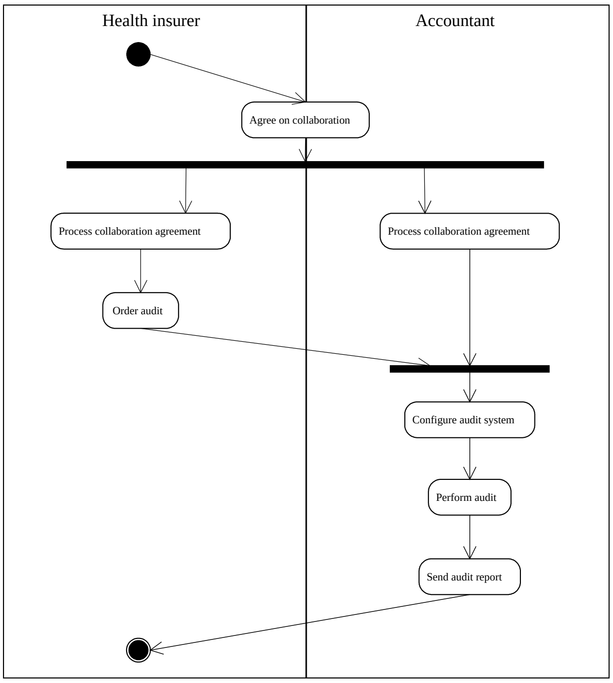

Figure C.1: Activities audit system at strategic, tactical and operational level

Figure C.2 gives the activity diagram for the activities at the tactical level, resulting in the configuration of the audit system. An explanation is given in Section 6.2.

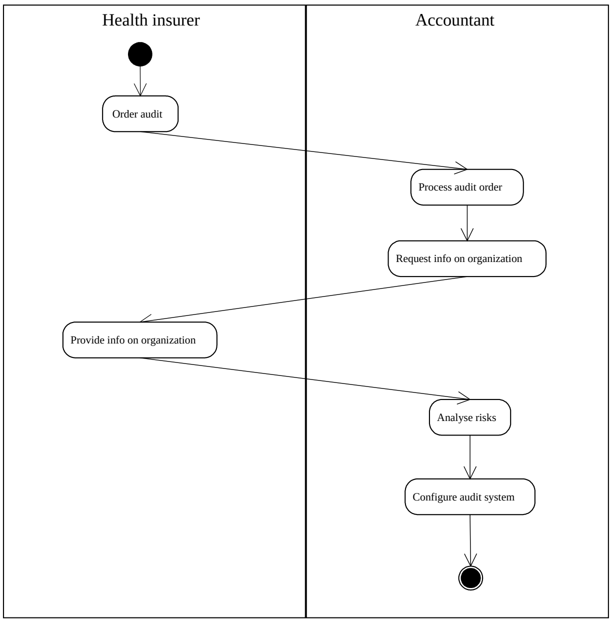

Figure C.2: Activities audit system at tactical level

The following figures provide activity diagrams for the operational audit process, as discussed in Section 6.4. Figure C.3 shows real-time data collection and consists of two processes. The first process is executed continuously and collects and converts data when received. The other process is triggered and performs the actual audit on the data. Figure C.4 shows the audit process with deferred data. The process is triggered and collects and converts all necessary data before it performs the actual audit.

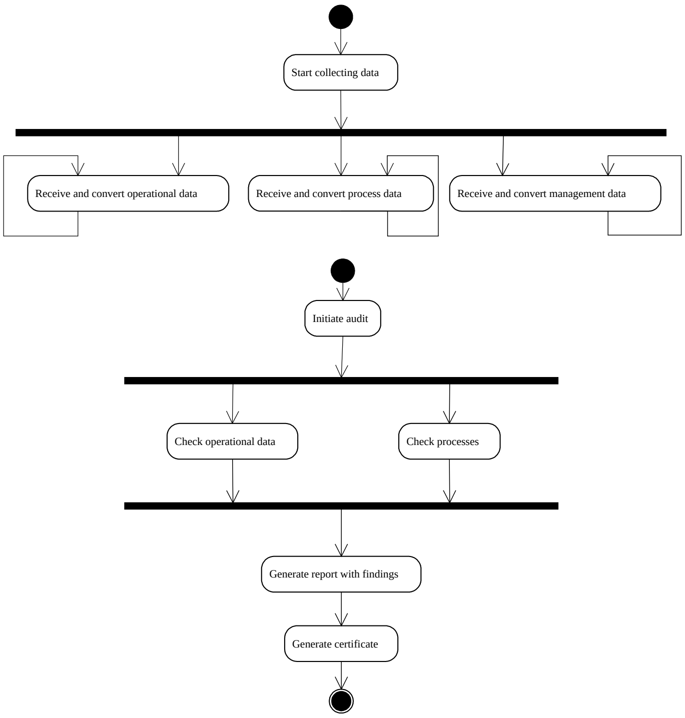
Start collecting data
Receive and convert operational data
Receive and convert process data
Receive and convert management data
Initiate audit
Check operational data
Check processes
Generate report with findings
Generate certificate

Figure C.3: Operational audit process: real-time data collection

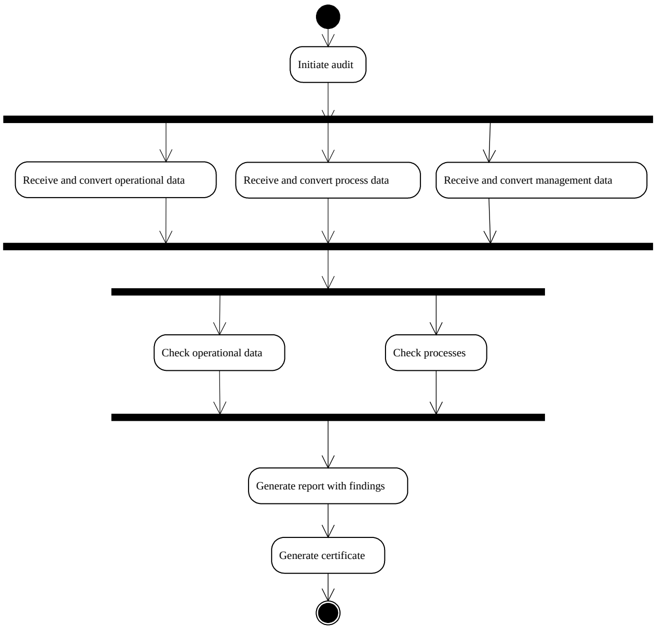
Initiate audit
Receive and convert operational data
Receive and convert process data
Receive and convert management data
Check operational data
Check processes
Generate report with findings
Generate certificate

Figure C.4: Operational audit process: deferred data collection

Appendix D

Overview relation audit system and elaborated premium process

Figure D.1 shows the connection between the audit system and the elaborated premium process, to show the specific data that is collected by the audit system.

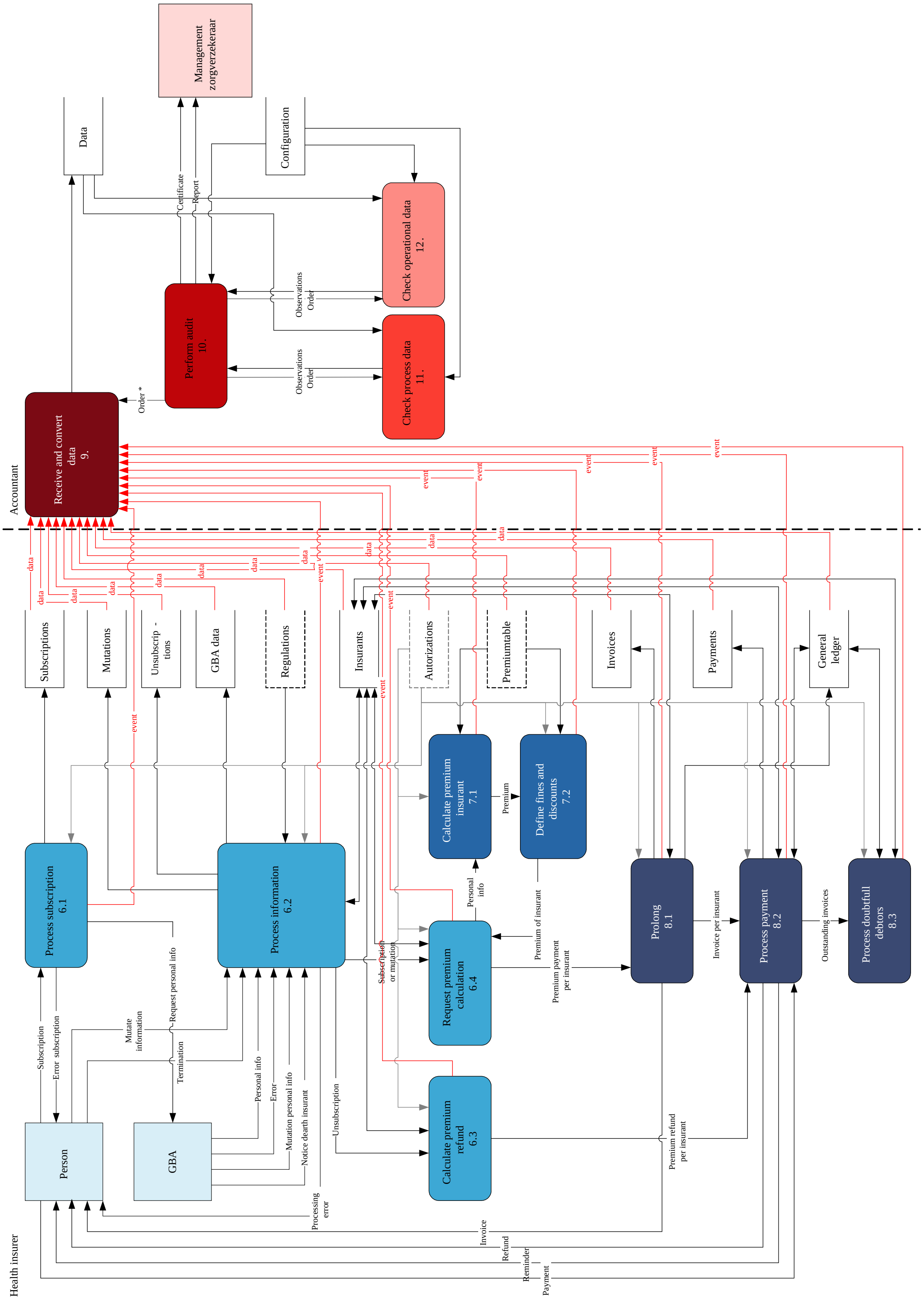

Figure D.1: Operational audit process and relation elaborated premium process