Architecture Evaluation Methods

AISA Project Report

Version: 2.0	Date: 2.5.2007
Author: Martin Hoffmann	Status: Final

Summary

This paper aims at presenting the current possibilities to evaluate enterprise and software architectures, focusing especially on performing an assessment mainly based on architectural descriptions. The essential research questions investigated in this paper are:

- What are the evaluation needs for architecture evaluation?
- What kind of architecture evaluation methods exist?
- Which needs do these evaluation techniques satisfy?
- What do the existing methods fail accomplish?

The studies of previous research resulted in the recognition that there is no methodology for enabling the enterprise architecture evaluation by considering the whole enterprise architecture. Therefore, methods, standards and measures which can be used to evaluate different concerns of enterprise architecture are presented. The evaluation techniques address the concerns of business, information, systems and technology separately. All of the introduced techniques have been developed or tested and validated in a practical environment.

In [1] evaluation needs have been identified by interviewing practitioners. Since it is also an aim of this paper to find approaches satisfying those needs the methods presented in this paper are mapped to those evaluation needs they address.

The methods suggested for the business architecture are:

- governance modelling
- business process modelling and simulation
- financial methods for assessing the value of an IT investment (prediction of expected benefits through IT investment)

The needs concerning the enterprise's information architecture were addressed by the evaluation of the corporate data model which is a structured conceptual model of the organisation's data entities and their relations. The suggested methodology was the *Moody's Framework*.

The systems architecture consists of software systems. A software system is described through software architectural artefacts. Therefore, the evaluation techniques suggested for the systems architecture are methods for software architecture evaluation (*questionnaires, scenario-based methods, design metrics, prototyping, mathematical modelling*). Since the infrastructure which allows the deployment of software applications is also part of the software system the underlying execution environment can be evaluated within the software architecture evaluation. The methods concerning the software system evaluation enable predictions regarding the whole system life cycle. Especially, characteristics, such as performance, cost, reliability and maintenance are essential characteristics in the enterprise architecture context. The suggested methods are able to assess these criteria.

The architecture evaluation depends strongly on conceptual models which are used to share and communicate the architectural knowledge among different stakeholders from different domains. Therefore, conceptual modelling standards are part of the evaluation methods or conceptual models belong to the evaluation input and are the basis for analysis and discussion about architectural decisions.

Contents

1 Introduction

The architecture of a system is the description of the system's structure and its behaviour. The system's structure embodies the components which can be active (e.g. human beings, applications, hardware components) and passive (e.g. communication channels, information storages) ones. The interaction of the components results in a certain system behaviour [2].

Every system has certain groups of stakeholders who have several interests towards the system. Usually, a system is implemented with a certain *vision* about the running system's task and the improvements to the current state achieved by the system. According to [3], the vision can be seen as a long-term purpose of the system. Since that vision is a final ultimate long-term achievement it is necessary to define *goals* which have to be achieved to achieve the final vision. That means the goals are a guideline to the final vision.

Goals are assessable because they are described though three dimensions [4]:

- content (direction of the goal)
- extent (scale of the goal)
- timing (timeframe of the goal)

Content means that the results, which are desired to be achieved, must be defined. The extent dimension quantifies the degree of the achievement and the time dimension fixes the period for achieving the goal.

In the paper [5], goals are seen as the stakeholders' success criteria. These goals are part of the system requirements. Since the architecture is the system's description, it must consider those requirements and it must be possible to assess the architecture regarding them. With accordance to [5], not all system requirements are considered by the architecture. The architecture is focusing on the realisation of so called *needs*. Needs differ from system requirements because needs are more stable over the system's life cycle. A need captures those concerns that will drive key decisions by the architect, such as decisions pertaining to performance, technology or cost drivers. Additionally, a need might be the abstraction or summarization of several detailed system requirements. Since the needs directly relate to the goals and the goals relate to the final vision, it is essential to evaluate the system's architecture regarding the realisation of the needs. The objectives of an architecture evaluation are:

- advancing and transferring architectural knowledge[6]
- identification of insufficiencies which are risks related to the needs [2]
- identification of design decisions and their contribution to the needs
- architectural decision making [6]
- choosing among several candidate architectures or design decisions [2]

The evaluation results are a useful basis for the system's improvement regarding the stakeholders' goals.

The fundamental evaluation process with its components is described, for example, in [7] and evaluation components of enterprise architecture (EA) evaluation are described in [8]. Several of these evaluation components have been investigated in the AISA project but still there seems to be a lack of research on evaluation methods. Therefore, it is necessary to identify methodologies for architecture evaluation in order to achieve the goals (of architecture evaluation) mentioned above. According to [5] architecture evaluation methodology itself must include the following tasks:

- Analysis of Needs, Goals and Vision
- Gather relevant documents and other artefacts related to the architecture
- Evaluate documentation against measures and score results
- Interpret results and identify architecture-related risks
- Documentation of results.

The scope of this paper is to identify architecture evaluation methods which can be applied for the evaluation of enterprise and software architectures, focusing especially on assessing the architectural descriptions regarding the identified needs. The requirements towards an enterprise and a software system are naturally different. Requirements towards software systems focus mainly on quality attributes like efficiency, reliability, security, and maintainability. A quality model for software systems is given in [9] and [10]. The stakeholders' goals towards an enterprise are more varying because of the rather huge complexity of enterprises. Quality attributes are important issues but also less tangible goals which are difficult to measure or predict, such as increased innovation, customer orientation, and market share. Especially, the evaluation of design decisions regarding strategic aims is quite challenging. Also the fact that the enterprise architecture embodies several architectures complicates an evaluation.

The paper is structured in the following way. The next section deals with evaluation needs which have been gained from practitioners through interviews [1]. Section 3 describes approaches for enterprise architecture evaluation concerning EA management processes and EA artefacts. In section 4, the most wide-spread approaches of EA management evaluation, *Maturity Models* and *IT-Business Alignment Models*, are presented. The approaches which address the evaluation of architectural artefacts of different views on enterprise architecture: business, information, software systems, and technology are discussed separately in sections 5-8. Software architecture evaluation methods are presented in the context of architecture evaluation of software systems. In section 9, the approaches are mapped on the needs described in section 2. The last section concludes the paper.

2 Architecture Evaluation Needs

Evaluation needs are essential stakeholders' concerns to the architecture which have to be evaluated. Since the focus of this paper lies on enterprise and software architectures, evaluation needs for both of these are investigated. The evaluation needs are derived from the goals of architecture evaluation and the stakeholder needs regarding the architecture.

Table 1 shows the evaluation needs for enterprise and software architectures. The needs have been identified from interviews with practitioners who are familiar with the stakeholders' concerns. In [1], it is stated that it is difficult for practitioners to directly name evaluation needs; usually certain concerns and needs for information trigger an evaluation. Therefore, the evaluation needs are derived from those triggers.

Table 1 Triggers for architecture evaluation [1]

Triggers for architecture evaluations	**Evaluation needs**	**Evaluation Targets**
A need for the documentation of good quality		
A need to produce architectural models and documentation that • can be quickly communicated and • are understandable by many different stakeholders • are cost-effectively kept up to date.	The evaluation of the quality of architectural documentation. A need to evaluate: - Policy: do policies (e.g architectural framework) exist for documentation and are they followed? - Intelligibility and usability: are documents easy to understand and use? - Accuracy: are documents truthful and factual? - Cost effectiveness of maintenance: how much effort is needed to keep models and documentation up to date? - Traceability between architectural documents: is there traceability between architectural documents?	Architecture documenta-tion (EA / SA)
A need to have organisation's business environment descriptions of good quality	The evaluation existence and quality of business descriptions (goals, strategy, company's operations) : • existence of business descriptions (e.g. goals, strategy, company's operations) • Accuracy: are the descriptions up to date?	Business architecture documenta-tion
A need to have information / data models of good quality	The evaluation of the quality the information / data models	Information / Data architecture
Change pressures in organisation		
A change need in the business or ICT (e.g. a	The evaluation and identification of the places affected by a change and effects in each	EA viewpoints

need to move from one solution to another)	architectural viewpoint.	
An observation that ICT-architecture do not correspond to company's business's requirements	The evaluation how the enterprise architecture should be changed by identifying what chances should be carried out in each architectural viewpoint.	EA viewpoints
The understanding of business and ICT environments		
A need to enhance the understanding of company's business/ICT	The evaluation of enterprise architecture from different aspects or against different factors e.g. the identification of overlaps.	EA viewpoints
A goal that ICT supports business	The evaluation of how business architecture is supported by other viewpoints (information, applications, infrastructure).	EA viewpoints
A need to enhance the understanding of responsibilities in the company	Identification and evaluation of responsibilities in company (for example: who is responsible for customer information).	Business architecture
A need to understand the state of the company's product portfolio and processes	The description and evaluation of business architecture related aspects.	Business architecture
A need to understand information managed in company	The description of major information entities and responsibilities in information management.	Information / Data architecture
A need to understand the state of the company's application portfolio	The description and evaluation of structures and components of application architecture.	Application architecture
A need to understand quality aspects relating to the company's application portfolio	The evaluation the application architecture against quality aspects and attributes e.g. the identification of overlaps.	Application architecture
A need to understand the current state of technical infrastructure	The description and evaluation of structures and components of technical infrastructure.	Technology architecture
Company management and process planning		
A need to make sure that organisational choices are suitable	The evaluation of organisational structures and operations: are those suitable or should those be changed.	Business architecture
The distribution of work	The evaluation of processes: identification of which tasks will be carried out by the company and which are dealt out to partners.	Business architecture
Business process planning	The evaluation of functionality of business processes: e.g. do processes correspond to company's strategy?	Business architecture
Management of architectures		
An observation that ICT-	The evaluation of how architectural principles	EA

architecture does not correspond to ICT-development projects' needs	or architecture descriptions should be changed.	viewpoints
An effort to drive investments to follow up architectural principles	The evaluation of if the investment corresponds and is suitable to the existing architecture and architectural principles.	EA viewpoints
A need to drive technical infrastructure investments to follow the architectural principles	The evaluation of if investments correspond to the principles.	Technology architecture principles
IT cost management		
A need to understand and manage costs relating to the company's application portfolio	The evaluation of financial aspects and factors relating to application architecture	Application architecture
A need to understand and manage costs relating to technical infrastructure	The evaluation of financial aspects and factors relating to technical infrastructure	Technology architecture
Architectural choices		
A need to find the best possible system solution and a need to understand the aspects relating the solution	The evaluation of the architectural solution: e.g. evaluation of • quality aspects (evaluation against quality attributes), • flexibility of solution, • the life cycle of solution, • suitability for the situation in question (e.g. is solution possible within available time, money and resources).	SA viewpoints (EA viewpoints)
An effort towards long-term technical solutions and need to argue for the long-term technical solutions	The comparison of a long-term and short-term solution.	EA / SA viewpoints

3 Enterprise Architecture Evaluation Methods

Today, more and more companies adopt enterprise architecture frameworks to cope with the changing environment and to improve their performance and competitiveness. Perhaps the most wide-spread frameworks are the Zachman Framework [11], TOGAF [12], FEAF [13] and DoDAF [14]. These frameworks typically combine different views of the enterprise, e.g. business, information, application, and technology architecture. These views should transfer knowledge about the organization towards involved stakeholder roles. Furthermore, they give a guideline for the necessary architectural documentation to describe the current enterprise architecture and also a future one. Unfortunately, there are many different concepts, modelling techniques, tool support, and visualisation techniques for every view. Consequently, there seems to be no coherent view on enterprise architecture. This fact also complicates the evaluation because it seems that there is no method which enables the assessment of the whole enterprise architecture. There are at least two main areas which can be evaluated regarding EA:

- enterprise architecture management and the management process
- architectural artefacts which describe the structure and behaviour of the EA.

Because there are no common EA evaluation methods, we decided to follow the structure given by most of the enterprise architecture frameworks [11], [12], [13] [14] and investigate techniques to evaluate architectural artefacts of the different views starting with the business architecture. Before that, however, concerning approaches to evaluate the EA management and management processes, a summary of the most wide-spread *Maturity Models* and *IT-Business Alignment Models* is presented.

4 Maturity Models and IT-Business Alignment

Existing enterprise architecture assessment techniques basically focus on the improvement of enterprise architecture management and the management process which means that new EA development targets are identified and development priorities are set. Therefore, enterprise maturity models and IT-Business-alignment evaluation are utilized. In the following the concepts of these methods are presented.

One of the first capability maturity models, Capability Maturity Model for Software (CMM), was developed by the Software Engineering Institute, Carnegie Mellon. It enables the assessment and the control of IT-related processes as well as the assessment of organization's development competence. According to [15], architecture maturity involves an organization's ability to organization-wide manage the development, implementation and maintenance of architectures on various levels – e.g. business, information, applications and infrastructure. That means architecture maturity focuses on the evaluation of the entire architecture organization which is responsible for architecture development. The architecture products they create, such as descriptions and models, are not addressed through those maturity models.

Most of the assessment models have been developed by consulting firms such as Gartner [16] and METAGroup [17], and federal agencies or organizations, such as the US Office of Management and Budget (OMB) [18], the US department of commerce (DoC) [19], and the National Association of State Chief Information Officers (NASCIO) [20]. These models generally work the same way as the early CMM. Basically, they use a number of criteria to assess architecture maturity. Typical criteria are, for example, process, governance, communication, technology, and business alignment. For each criterion five maturity levels exist and they are provided with a description of aspects. The individual level of maturity for each of the criteria is based on questionnaires which are answered by certain stakeholder groups. The maturity models differ in the amount of criteria which are investigated. However, no matter which model is applied, they all support the identification of insufficiencies and areas of improvement in the enterprise architecture development process.

Another approach to assess the EA management and development processes is IT-Business alignment. There is a general agreement what alignment entails: the fit between business strategy, IT strategy, organizational structures and processes, and IT structures and processes [21]. The aim of alignment is for IT activities to support those of the entire business [22].

Several alignment assessment models have been constructed. One well-known model is Luftman's strategic alignment assessment model which presents an approach for determining a company's business-IT alignment based on six criteria: communications, competency/value measurements, governance, partnership, skills, as well as scope and architecture [21]. This last criterion is used to evaluate IT maturity. According to [21], each of these six variables is assigned five levels of alignment. The model provides a short description of the aspects of each level. The level of alignment for each individual variable is determined by the answers to 6 or 7 questions. The model also describes the process of conducting an alignment assessment. Luftman created this alignment assessment model based on his extensive research and practical experience.

The Chief Information Officer (CIO) Council, a consortium of US Federal executive agency CIO's, developed an architecture specific alignment and assessment guide [23]. This guide describes a process which consists of three phases, the select phase, control phase, and evaluate phase. First, the select phase entails assessing business alignment; whether and to what degree a proposed investment aligns with business strategy. Second, in the control phase the technical alignment is assessed on how well the technology of investments aligns with the infrastructure architecture. Finally, the third phase evaluates both the architectural products and the architecture development process itself.

5 Business Architecture Evaluation

According to TOGAF [12], the main aspects of the business architecture are:

- Business goals and objectives
- Business functions
- Business processes
- Business roles
- Business data model (the data model is considered in Section 6)

They all have to be documented in an appropriate manner which enables the analysis and evaluation.

Since the business architecture transfers this essential knowledge about the organization to all kinds of stakeholders like business users, business analysts, and technical developers it is strongly relying on conceptual modelling to be understandable for different domains.

5.1 Business Governance Modelling

Vision, goals, objectives and other aspects of the organization's governance determine the strategies which result into actions to transform the enterprise's as-is status into the desired to-be status. Since the governance is the foundation for the organizational structures, processes and behaviour it should be documented within the models describing EA. Usually, enterprises only capture the means to achieve goals in models [24]. That makes the traceability, analysis and evaluation of goals rather difficult.

Modelling the corporate governance would bring several benefits to the organization:

- vision, goals, objectives are made explicit
- transparency of transformation drivers [24]
- tracing of decisions and responsibilities
- basis for analysis and evaluation (conflicts, improvement, level of fulfilment)
- basis for planning and changing strategies and processes (linking *why*-knowledge to *how* [24])

One of the few notations that can be used for modelling the business governance is the Business Motivation Model (BMM). It is a meta-model of concepts for modelling the business governance. It has been standardized by the Object Management Group (OMG) in August 2006. Its purpose is to capture business motivation and intentionality by providing a scheme to develop, communicate and organize corporate governance [25]. Central element groups in the BMM are: Means, Ends, Influencer, Potential Impact and Assessments. These central elements are further refined into elements such as *Visions*, *Desired Results*, *Goals*, *Objectives*, *Missions*, *Course of Action*, and *Internal* or *External Influencers*.

The model's core concept is the connection of Means and Ends. Ends include the elements Vision, Goal, and Objective. Means refer to the concepts of Mission, Strategy, and Tactic. BMM is based

on the refinement of vision into goals and objectives, and a mission into strategies for approaching goals, and tactics for achieving objectives. The model also considers the fact that business needs to take into account the numerous influencers that can have positive or negative impact on the business. The assessment whether an influencer is strength/opportunity or weakness/threat is usually gained from the Strength-Weakness-Opportunity-Threat Analysis (SWOT) [26]. The BMM in Figure 1 illustrates the relation between governance aspects. BMM supports the understanding of the relations between intentional aspects of the governance level and also their relation to actions and processes performed by the organisation.

Currently, there seem to be no methods for systematic goal analysis for the EA evaluation which have been applied in a practical or industrial case study. Regarding goal analysis it might be possible to apply the approaches of the goal-oriented requirements engineering, such as Mylopoulos [27], i* [28], and EEML [29], to gain knowledge if goals are conflicting, complete and relevant. However, that is more an idea of further research than a suggestion for the practical use.

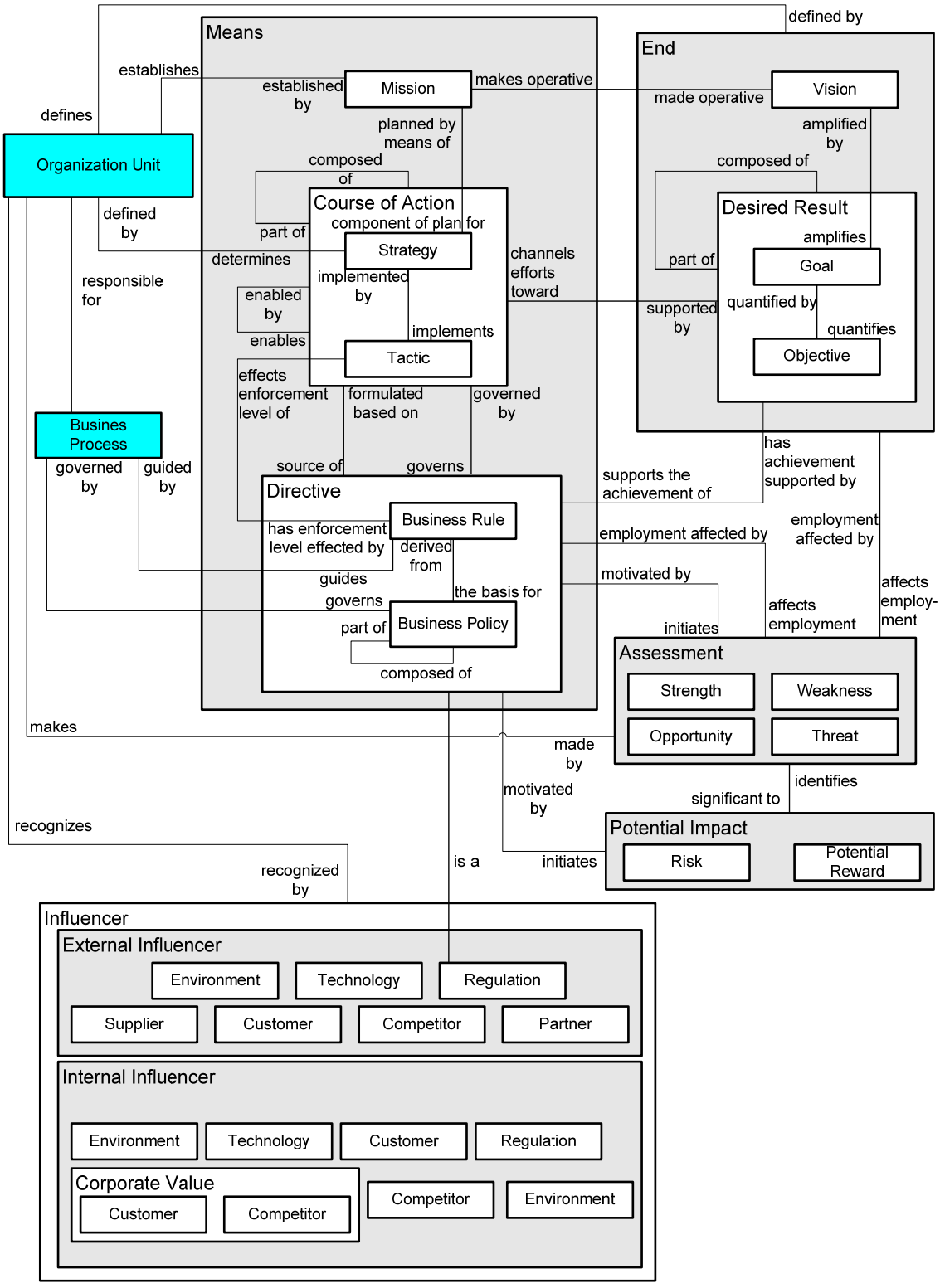

Figure 1 BMM of relations between governance aspects

5.2 Business Process Modelling and Simulation

A quite common means to gain a competitive advantage, regarding costs or innovation, is the optimization of an organization's business processes. The optimization embodies the assessment of necessary infrastructure and applications, and comparison of expected benefits [30]. Business process modelling and simulation are the approaches to achieve the optimization of processes [31].

5.2.1 Business Process Modelling

Business process modelling is the visualization of processes regarding relationships, dependencies, and effects between processes and their activities and resources. This visualization increases the understanding about the processes and supports the validation and improvement for many stakeholders [31]. Business process modelling aims at clarifying the organization's processes to its employees. Usually, even the documentation of processes discloses redundancies and points of improvement. According to [30], 80% of process advancements are achieved by modelling the current status.

Business process modelling consists of the following phases [30]:

- examining and modelling the organizational structure
- examining and modelling the existing business processes (*as-is state*)
- creating a base of the company's business processes
- verifying business processes
- analysing weak points
- modelling advanced business processes (*to-be state*)

There is several business process modelling approaches available. The three common approaches are:

1. Event-Driven Process Chain (EPC)
2. Business Process Modeling Notation (BPMN)
3. Unified Modeling Language (UML)

In the following EPC and BPMN approaches are investigated in more detail. Processes modelled with these two languages can be executed which is essential regarding business process simulation and implementation.

Event-Driven Process Chain (EPC)

The method of event-driven process chain (EPC) has been developed within the framework of *Architecture of Integrated Information System* (ARIS) [32] and is used by many companies for modelling, analyzing, and redesigning business processes. ARIS divides an organization's processes into separate views to reduce the complexity. The views are functions, data, organization, and control [32]:

- The *Data View* contains events and statuses. Events such as *customer's order received* or statuses, like an article description, are data objects. Chen's Entity-Relationship model [33] was adopted into the ARIS framework to create the organization's data model.
- The *Function View* describes the activities to be performed by the process, the individual sub functions, and their relationships.

- The *Organization View* represents the organizational structure. This includes the relationships between organizational units, between employees and organizational units, and employees and roles.
- The *Control View* links functions, organization and data, thus integrating the design results of the different views.

The various elements are connected into a common context by the control flow. The resulting model is the *Event-Driven Process Chain* [34]. The EPC is based on the concepts of stochastic networks and Petri nets. A basic EPC consists of the following elements:

- *Functions* are active elements representing the activities within the company.
- *Events* are created by processing functions or by actors outside of the model
- *Logical operators* (AND, XOR and OR) connect functions and events

Figure 2 illustrates the EPC elements exemplarily on an order process.

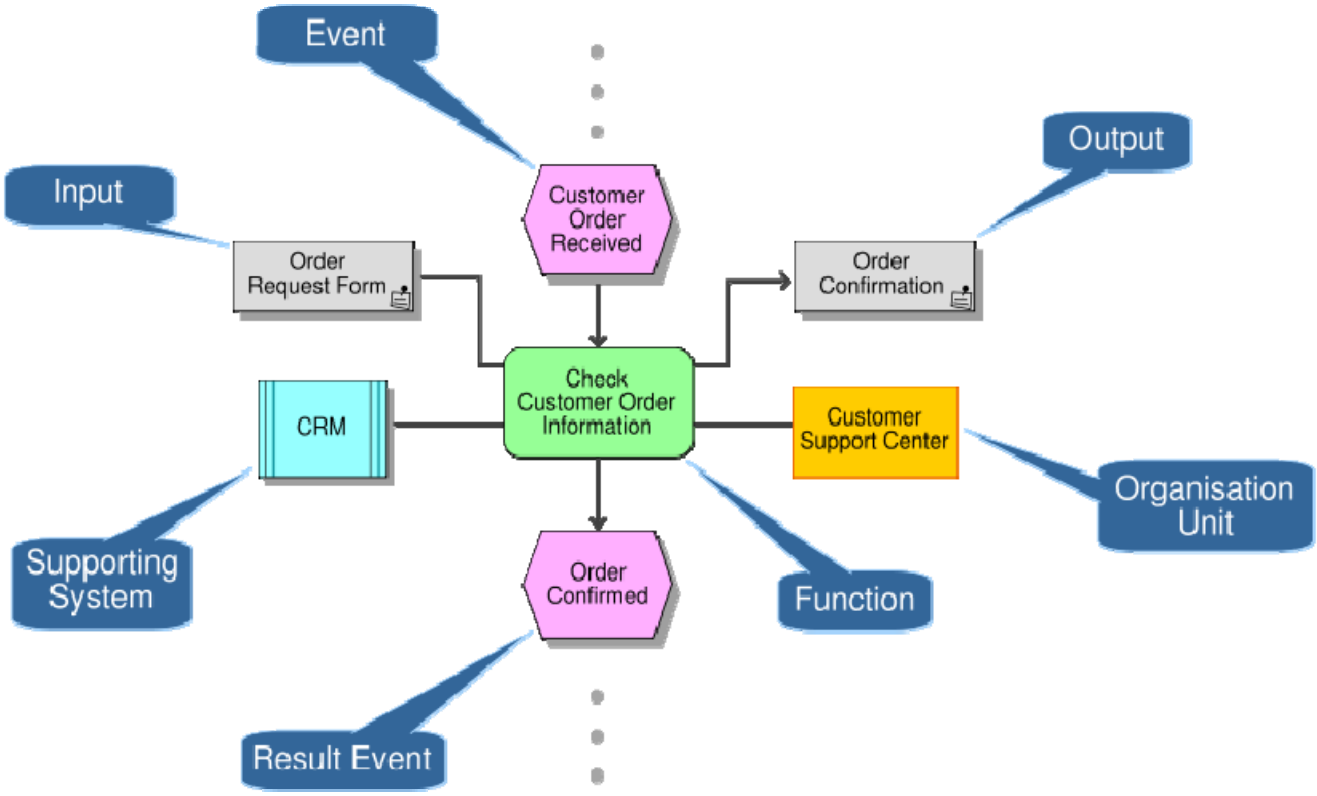

Figure 2 Elements of EPC

BPMN – The Object Management Group (OMG) Standard

The *Business Process Management Initiative* (BPMI) has developed a standard *Business Process Modeling Notation* (BPMN). The BPMN 1.0 specification was released to the public in May, 2004.
.

The primary goal of BPMN is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes [35]. Thus, BPMN is meant to create a standardized bridge for the gap between the business process design and process implementation. BPMN includes four basic categories of elements [35]:

1. Flow Objects
2. Connecting Objects
3. Swimlanes
4. Artifacts

Actually, the first category is the most important one. *Events*, *Activities*, and *Gateways* (represent Decisions) belong to the category of Flow Objects. These elements correspond to the EPC's elements Event, Function and Logical Operators. Connecting Objects include Sequence Flow, Message Flow, and Association which are used for relating the Flow Objects.

The concept of *Swimlanes is used* as a mechanism to organize activities into separate visual categories in order to illustrate different functional capabilities or responsibilities. *Artifacts* allow adding extra context to the diagram. Therefore, BPMN defines Data Object, Group, and Annotation. Figure 3 illustrate a reservation process modelled with BMPN.

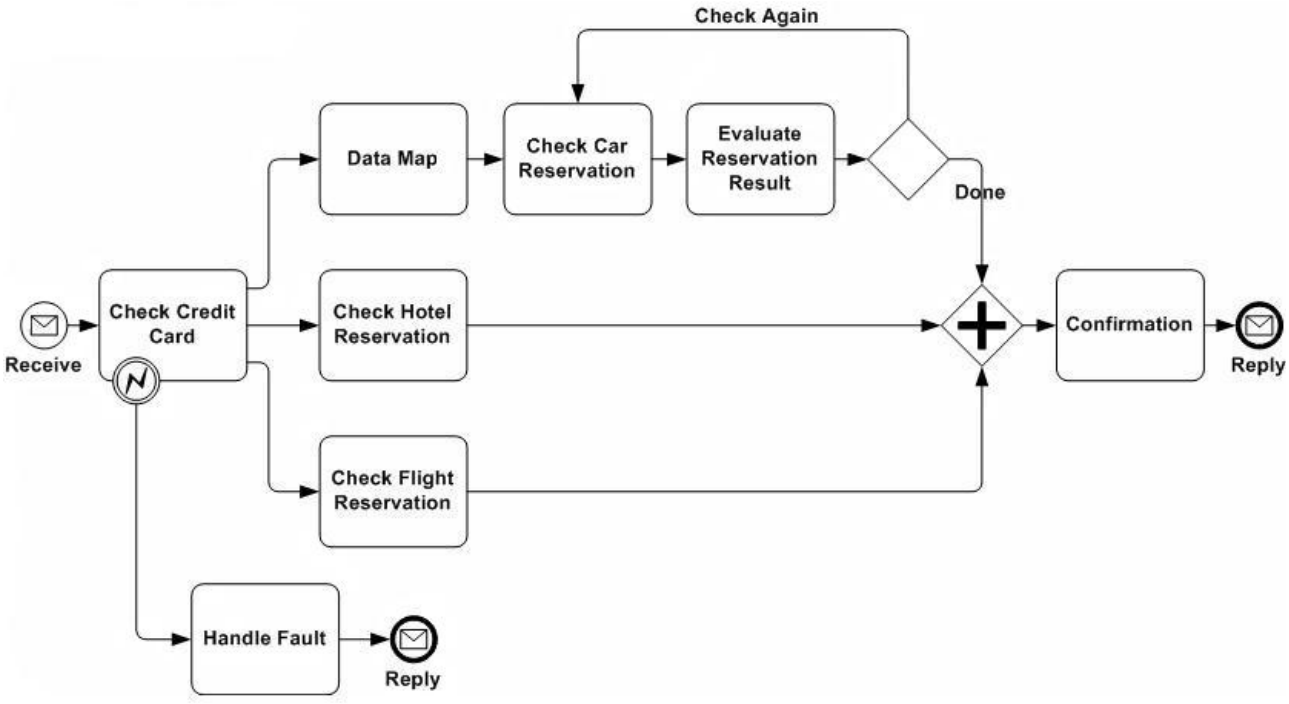

Figure 3 Reservation process in BPMN

5.2.2 Business Process Simulation

While modelling is the visualization of business processes, simulation brings them alive. On the one hand, it is possible to evaluate the current processes (*as-is* state) regarding costs, performance and to analyse the simulation data referring optimization. On the other hand, dynamic simulation is a way to analyze *what-if* scenarios, obtain cost and performance predictions, and validate processes [31]. The predictions, gained from the simulation, support the decision making regarding organizational change and future investments.

In the previous section, the two common description graphical description languages EPC and BPMN were described. The advantage of both of these languages is that the described processes can be executed for simulation as well as for implementation purposes. Because EPC is used within the ARIS framework [32] the created models can be simulated within the ARIS environment with three analysis tools: ARIS Simulation, ARIS ABC (Activity Based Costing), and ARIS BSC (Balanced Scorecard). The simulation with ARIS provides information about the executability of processes, processes' weak points and resource bottlenecks [32] [30].

The process models in BPMN are well understandable by human beings but not by computers. Therefore, BPMN has to be translated into an executable language, such as Business Process Execution Language for Web Services (BPEL) [36] which is emerging as a de-facto standard for implementing business processes on top of web services technology. Numerous platforms support the execution of BPEL processes. Some of these platforms also provide graphical editing tools for defining BPEL processes. However, these tools directly follow the syntax of BPEL which does not support the level of abstraction that BPMN is using during the analysis and design phases of the business processes. BPMN has attained some level of adoption among business analysts and system architects as a language for defining business process blueprints for subsequent implementation [37]. Meanwhile, BPMN is already supported by more than 30 tools, for example Appian Enterprise 5 Business Process Management Suite and BizAgi.

5.3 Financial methods for assessing the business value of IT investments

Organizations use several measures to assess business value, such as return on invest (ROI), net present value (NPV), internal rate of return (IRR), payback period, and economic value added (EVA). According to [38], these measures have five main disadvantages regarding their utilization to measure the business value of IT.

- There are too many measures available and within a single organization different groups use different measures; furthermore, some measures have multiple interpretations which lead to inconsistency.
- These measures generate a value which leads to a wrong credibility because the value is actually based on assumptions and the value itself is only a prediction for the estimated benefit.
- These measures do not take intangible benefits, such as customer satisfaction, into account. Since it is difficult to measure intangible benefits they are completely ignored.
- The financial measures only estimate the direct benefit of an investment but they are not able to calculate further future benefits or opportunities.
- Perhaps the biggest flaw in most financial measurements is the underestimation of risks or even the failure to incorporate any risk at all.

Since, measuring the value of IT-enabled business change will be critical to almost every organization as technology becomes embedded in virtually every business process [38], more efficient measurement tools are needed. In this report, four methodologies which have been developed to overcome the problems of the standard financial measures are addressed:

1. Business Value Index (BVI)
2. Total Economic Impact (TEI)
3. Val IT
4. Applied Information Economics (AIE)

The Business Value Index [39] is a method which was developed by Intel's IT organization. The method was first applied in 2001. Basically, BVI supports the prioritization of investment options. Tangible as well as intangible can be measured. BVI is a composite index of factors that impact the value of an IT investment. Basically, BVI assesses IT investments along a three dimensional vector consisting of the dimensions:

1. IT business value (that is, impact to Intel's business) represents the tangible and intangible benefits. There are some projects which have significant business value (e.g., responding to a competitor's threat or customer's satisfaction) but may not be financially attractive. In these cases value for the organization is captured by this dimension.
2. Impact to IT efficiency measures a projects impact on the IT organization. IT organizations are increasingly developing enterprise architectures, establishing standards, and acquiring core competencies in key skill areas to reduce costs and become more agile [35]. IT efficiency is Intel's measure to assess a project's conformance to the established architecture. A project that does not conform organization's standards and frameworks will be more costly to implement and support and will also entail greater risks [35].
3. Financial attractiveness of an investment is determined using at least three financial metrics; NPV, IRR, and payback period together.

In Appendix 1, an example of using the BVI is shortly described.

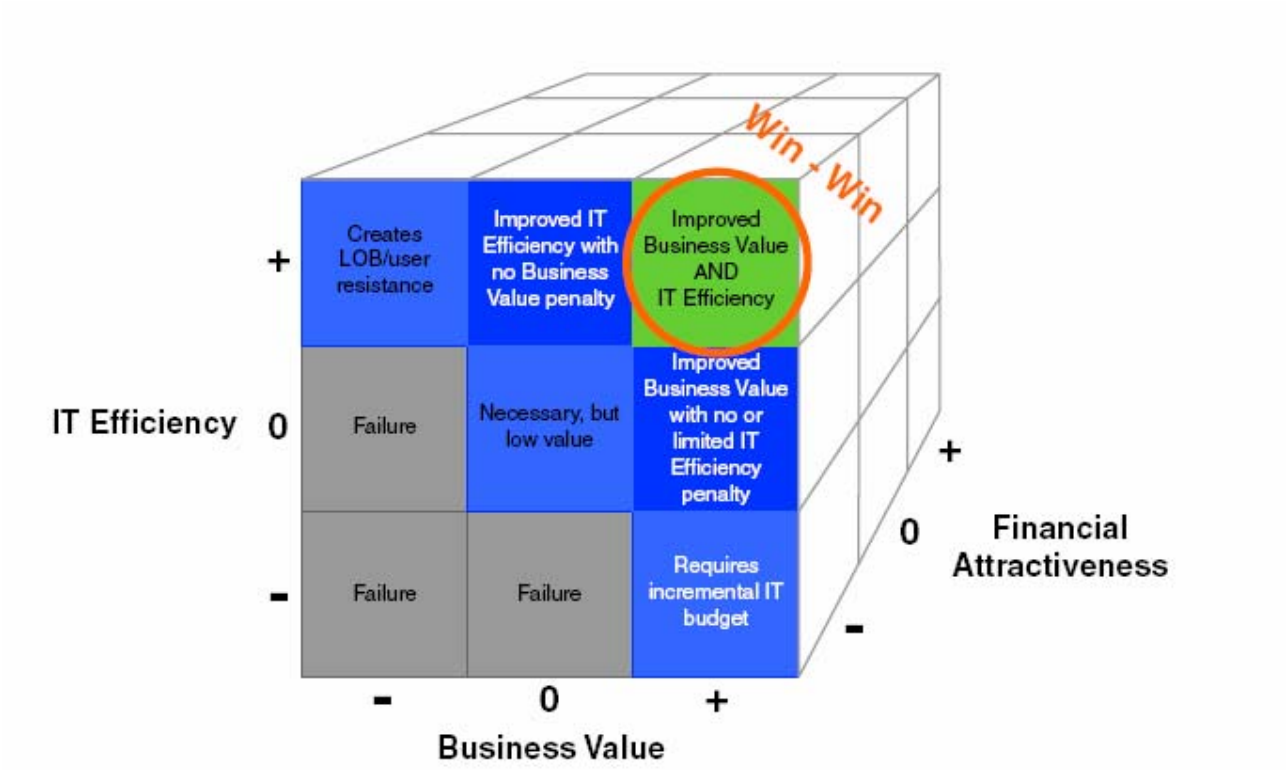

Figure 4 BVI dimensions [39]

The Total Economic Impact (TEI) is Forrester's methodology for assessing IT investments. TEI systematically looks at the potential affects of technology investments across four dimensions [38]:

- Cost — impact on IT.
- Benefits — impact on business.
- Flexibility — future options created by the investment.
- Risk — uncertainty.

The TEI cost dimension considers the changes in IT costs compared with maintaining the current status. Cost changes can be seen as the required investment to bring this new initiative, application, or technology online [38]. These cost changes are usually higher during development or implementation phases and then potentially decrease over time. The impact on IT can be positive if costs are decreasing or negative if costs are increasing [38].

TEI's benefit dimensions regards the impact of IT investments on the non-IT departments. Usually, the initial implementation requires changes to personnel or behaviour in the effected user departments. Users have to be taught using the new systems and in the beginning the efficiency of the departments using new IT systems might suffer. Then the initial benefit is rather small but on a long-term view that will be compensated by an improved productivity gain, or other positive impact.

Future options, or flexibility, can be looked at as the value of the option to take a second or third action in the future [38]. Form this point of view; it is similar to a financial purchase option. Investing in additional infrastructure in excess of today's needs, for example, can enable the

deployment of future applications. The standard measurements are not able to estimate the benefits of these applications but their right to take these actions in the future still has value to the organization and the scale of that value should be monetized and communicated [38].

In TEI, the risk analysis translates the initial estimates for cost, benefits, and future options into a range of potential outcomes. Once this range has been determined, by either adjusting the final estimates or by evaluating the effect of risk on the individual components of the cost and benefits, an expected value for this range of possible outcomes can be determined [38]. That means considering the potential risk for the three dimensions Flexibility, Benefits, and Costs a minimum and maximum benefit can be estimated. According to [38], it is called “risk-adjusted ROI”.

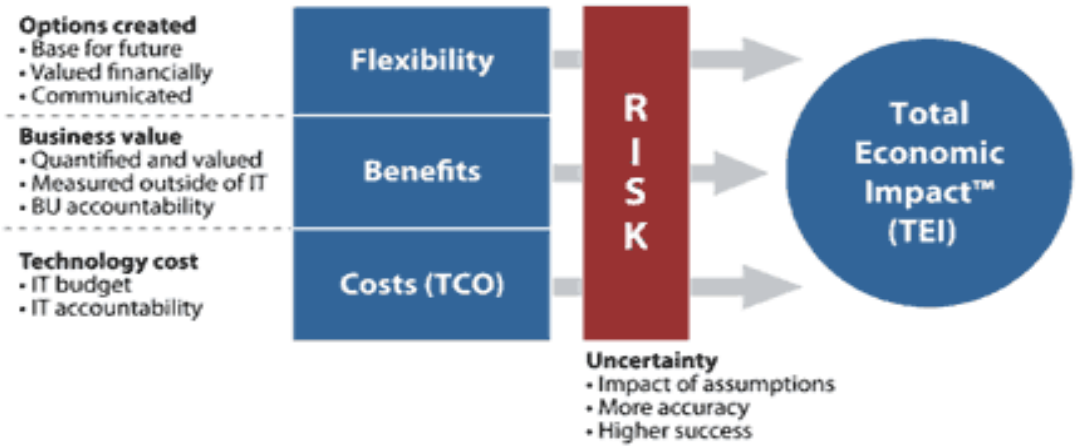

Figure 5 Total Economic Impact (TEI) [38]

A frame work for measuring the IT value, Val IT, has recently been released by the IT Governance Institute (ITGI). It is a governance framework that consists of a set of guiding principles, and a number of processes conforming to those principles that are further defined as a set of key management practices. Val IT, illustrated in Figure 6, comprises three key processes (including 41 key management practices) [40]:

1. Value governance, contains 11 key management practices which cover
    - The establishment of a governance, monitoring and control framework
    - The provision of strategic direction
    - The definition of investment portfolio objectives
2. Portfolio management, includes 14 key management practices covering
    - The establishment and maintenance of resource profiles
    - The definition of investment thresholds
    - Evaluation, prioritization and selection, deferral, or rejection of new investments
    - Management of the overall portfolio
    - Monitoring and reporting on portfolio performance
3. Investment management, consists of 15 key management practices which cover
    - The identification of business requirements
    - The development of a clear understanding of candidate investment programs
    - The analysis of alternatives

- Program definition and documentation of a detailed business case, including benefits details
- Assignment of clear accountability and ownership
- Management of the program through its full economic life cycle.

Since the Val IT framework is rather new there is not much practical experience in applying the framework so far. Much of the framework's content was validated by the Dutch financial services firm ING [41].

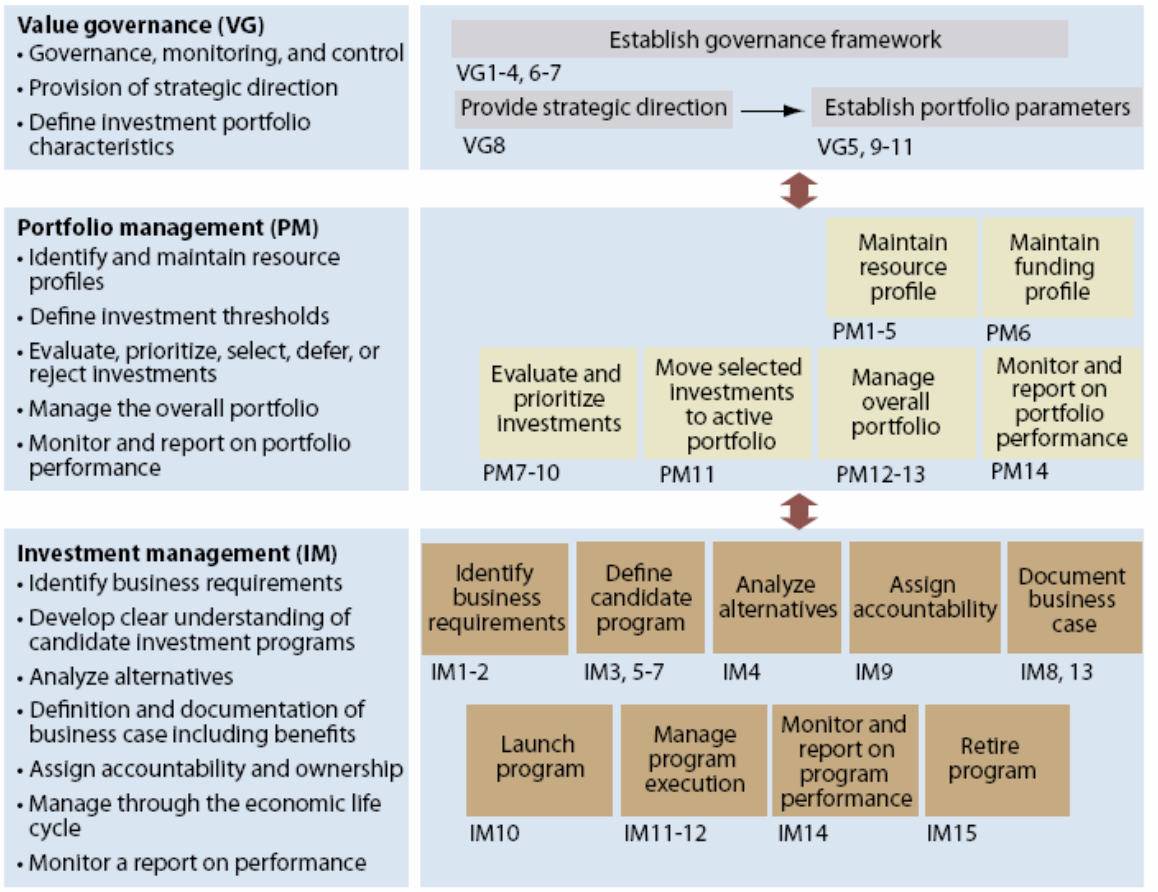

Figure 6 Val IT framework with its key processes and the management practices [42]

The Applied Information Economics (AIE) is a practical application of mathematical and scientific methods to the IT and business decision process [43]. It includes methodologies from economics, operation research, portfolio theory, software metrics, decision/game theory, actuarial science, and options theory into a precise, highly quantitative methodology for assessing IT investments. AIE can be applied across the enterprise to solve some of its most perplexing problems, including the following [43]:

- Using mathematical models to improve cost/benefit analysis (CBA) for better decisions at all levels of IT

- Developing financially-based quality assurance measurements to insure that the implementation of IT decisions are effective
- Developing a strategic plan for information systems based on identifying the best opportunities for economic contribution by information systems.

AIE embodies a number of basic techniques [43], such as *unit of measure* definitions, calculation methods for the value of information, methods for modelling uncertainty in estimates, and treating the IT investment as a type of investment portfolio. These methods are also used by financial services firms to create financial products and they are also used by insurance companies to calculate premiums. The AIE's key methods are:

- Unit of measure: IT investment also includes intangible or not measurable factors, such as strategic alignment, customer satisfaction, or employee empowerment. Mostly these factors only seem to be immeasurable because they are ambiguously defined. AIE removes this type of ambiguity by focusing on definitions that can be expressed in units of measure.
- Uncertainty analysis: According to [43], all investments have a measurable amount of uncertainty or risk. AIE is able to quantify the risk of a given IT investment, and compare its risk/return with other non-IT investments. AIE quantifies uncertainties with ranges of values and probabilities.
- Calculation of Economic Value of Information: The basic assumption of AIE is that the value of information can be calculated [43]. Since information reduces uncertainty, it supports decision making. The improved decision making results in more effective actions and those actions might lead the higher profits or the achievement of other goals. The relation between the information and its impact on the profit or goal can be expressed in value.
- IT Investments as an Investment Portfolio: AIE also uses the methods of Modern Portfolio Theory (MPT) and considers the set of an organization's IT investments as another type of investment portfolio [43].

Even though AIE was developed over a decade ago, it has not gained widespread use. The method is mostly applied in the government sector.

To conclude the financial method section, a brief comparison between the methods is presented in Figure 7.According to [38], BVI is the simplest method; especially organizations with no history of applying value methodologies might find BVI easier to implement. It is well-documented and more qualitative in its assessments of benefits and risks although it does incorporate standard financial measures.

TEI adds more rigor around quantifying intangible benefits, risk, and the value of flexibility or future capability resulting from IT investments. Organizations that are risk averse or that plan on making large investments in infrastructure or other capabilities might benefit from using TEI [38].

Val IT takes a governance approach. However, due to its relative immaturity, it is suggested by [38] to be prudent to wait for the methodology to be more fully built out and more experience gained with its use, although much of the Val IT methodology has been in use by ING for a number of years.

Organizations requiring more quantitative rigor may adopt AIE. With its mathematical, statistical, and economic techniques AIE provides investment decision-makers with a high degree of confidence in its results. However, there is a steep learning curve associated with it and it requires significant expertise [38].

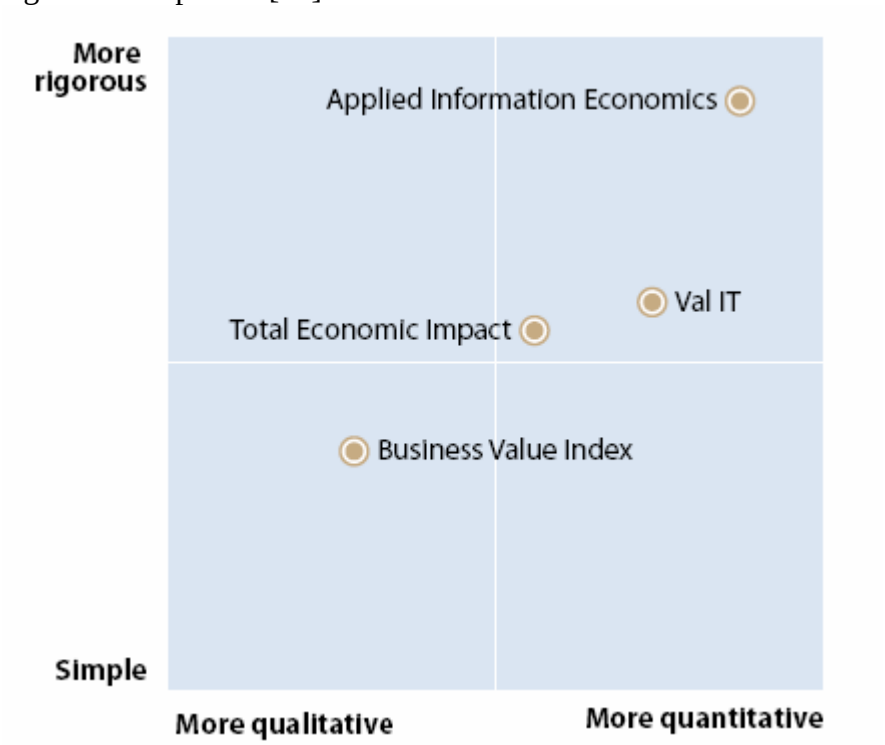

Figure 7 Comparison of methods regarding difficulty of usage and accuracy [38]

6 Information Architecture Evaluation

The Information architecture is a high-level model of information which an organization needs in order to make decision referring the future and required changes and also to perform its operative processes [44]. Storage, sharing and the integrity of information and data within the organisation is performed by Information Systems (IS). IS usually consists of two aspects: the *data architecture* and *data presentation* for operational issues. However, the role of data is much more essential for the IS because the data consists of all available information of the enterprise and the relations between different information. The information is necessary to perform the enterprise's processes. Therefore, the way the data is organized in the IS affects the process in a positive or negative way.

The Data architecture defines how data is stored, managed, and used in a system [45]. It provides criteria for data processing operations that make it possible to design data flows and also control the flow of data in the system. The data warehouse is a common approach for storing and analyzing the data which is created within or outside an organization. A data warehouse consists of a database management system (DBMS) with several databases. The data warehouse or any other database system implements a *corporate data model* [46] which is the relevant issue because it is a conceptual and structured model of the organization's data. The quality of the IS depends on the conceptual data models' quality but there is a lack of quantitative methods to assess the quality of data models. Several frameworks for evaluating a data model's quality have been suggested in [47], [48], [49], and [50]. However, most of these frameworks suggest criteria that may be used to evaluate the quality of data models but an evaluation that is based only on criteria is quite difficult because criteria may be interpreted differently [51]. In this report, one framework which was used already in several companies to evaluate their data models' quality is introduced in the following.

6.1 Moody's Framework for Evaluating and Improving the Quality of Data Models

While studying the previous research, only the *Moody's Framework* for the evaluation of the quality of data models (Entity-Relationship diagrams) was found. The framework was developed in practice and has been applied on a wide range of organizations [51]. The main components of the framework are summarized by the Entity Relationship model shown in Figure 8. The framework defines necessary quality factors which are illustrated in Figure 9. Furthermore, also the assigned stakeholder roles are shown for each of the quality factors. To assess these quality factors the framework embodies a number of evaluation methods, which in some cases are measures (e.g. data model complexity) and in other cases are processes for carrying out the evaluation (e.g. user views). In the following these methods and the quality factor they refer to are presented.

6.1.1 Evaluating the Completeness of NN

The data model is considered as complete if it contains all information required to meet user requirements. This corresponds to one half of the 100 % principle that the conceptual schema should define all static aspects of the Universe of Discourse [52]. Completeness is the most important quality factor because if it is not satisfied, none of the other quality factors matter. An inaccurate or incomplete data model results in a IS which will not satisfy users, no matter how well designed or implemented it is [51].

Generally, completeness can be checked by checking that each user requirement is represented somewhere in the model, and that each element of the model corresponds to a user requirement [53]. Therefore, completeness can only be evaluated in cooperation with business users. The result of completeness reviews will be a list of elements (entities, relationships, attributes, business rules) that do not match user requirements [51].

The Moody's measures for completeness consider mismatches with respect to user requirements. The given metrics should be considered during the review process, so that the model exactly matches user requirements. The completeness metrics are introduced in Appendix 2.

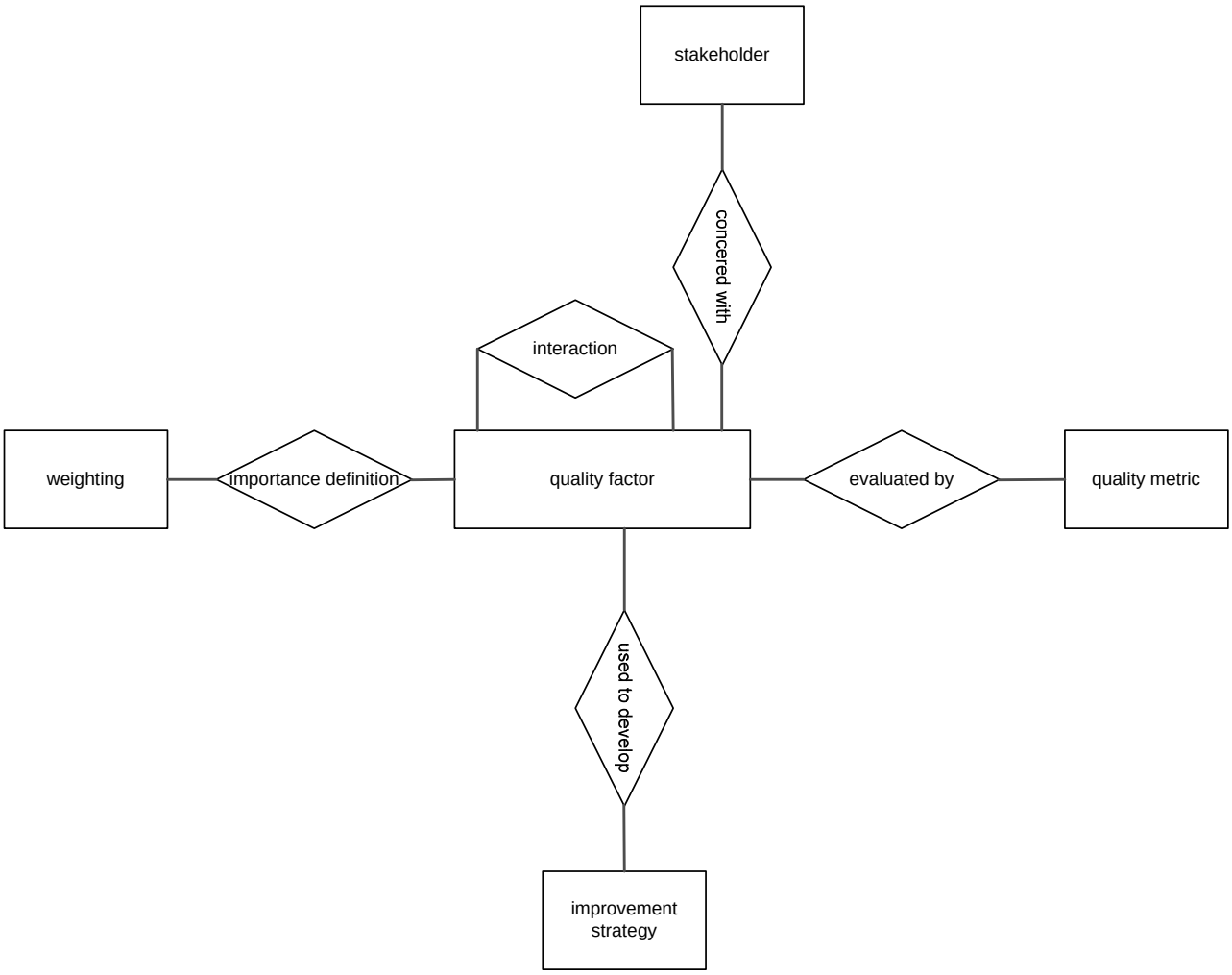

Figure 8: Components of Moody's Framework (Chen notation)

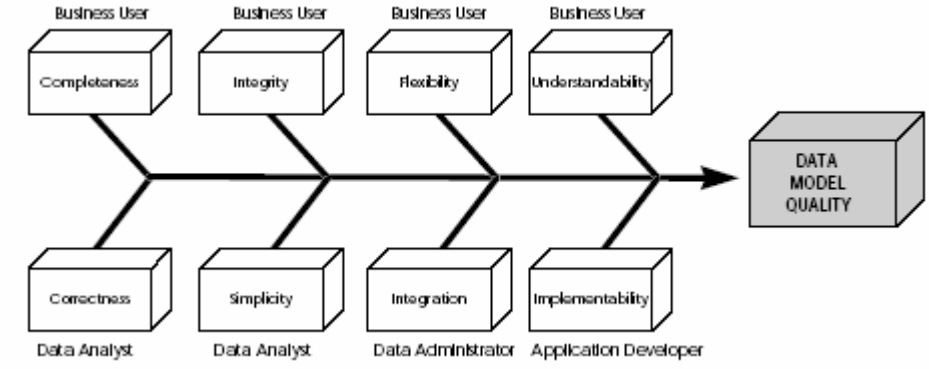

Figure 9 Data Model Quality Factors [51]

6.1.2 Evaluating the Integrity of NN

Integrity is defined as the extent to which the business rules (or integrity constraints) which apply to the data are enforced by the data model [51]. Integrity corresponds to the other half of the 100%

principle that the conceptual schema should define all dynamic aspects of the Universe of Discourse [52]. Business rules define what can and can't happen to the data. Business rules are necessary to maintain the consistency and integrity of data stored, as well as to enforce business policies ([54], [55]). The data model should include all rules which can be applied on the data to ensure they are enforced consistently across all application programs [52].

Like completeness, integrity can only really be evaluated with close participation of business users [51]. The rules represented by the data model may be verified by translating them into natural language sentences. Users can then verify whether each rule is true or false.

The proposed quality measures for integrity take the form of mismatches between the data model and business policies. The Integrity metrics are given in Appendix 2.

6.1.3 Evaluating the Flexibility of NN

Flexibility is defined as the ease with which the data model can cope with business change [51]. The objective is for additions and/or changes in requirements to be handled with the minimum possible change to the data model. Lack of flexibility in the data model can lead to:

- Maintenance costs: of all types of maintenance changes, changes to data structures and formats are the most expensive. This is because each such change has a "ripple effect" on all the programs that use it.
- Reduced organisational responsiveness: inflexible systems inhibit changes to business practices, organisational growth and the ability to respond quickly to business or regulatory change. Often the major constraint on introducing business change

The evaluation of the quality factor Flexibility is complicated by the inherent difficulty of predicting what might happen in the future. Flexibility evaluation requires identifying what requirements might change in the future, their probability of occurrence and their influence on the data model. However, no matter how much time spent thinking about what might happen in the future, such changes remain hard to anticipate.

The proposed flexibility metrics focus on areas where the model is potentially unstable, where changes to the model might be required in the future as a result of changes in the business environment. The purpose of the review process will be to look at ways of minimising impact of change on the model, taking into account the probability of change, strategic impact and likely cost of change. A particular focus of flexibility reviews is identifying business rules which might change. The flexibility metrics are introduced in the Appendix 2.

6.1.4 Evaluating the Understandability of NN

Understandability is defined as the ease with which the data model can be understood [51]. Business users must be able to understand the model in order to verify that it meets their requirements. Similarly, application developers need to be able to understand the model to implement it correctly. Understandability is also important in terms of the usability of the system. If users have trouble understanding the concepts in the data model, they are also likely to have difficulty understanding the system which is produced as a result. The communication properties of the data model are critical to the success of the modelling effort. [51].

Understandability can only be evaluated with close participation of the users of the model such as business users and application developers. In principle, understandability can be checked by checking that each element of the model is understandable. However, this is practically difficult because users might think they understand the model while not understanding its full implications and possible limitations from a business perspective.

The proposed measures for understandability take the form of ratings by different stakeholders and tests of understanding. The purpose of the review process will be to maximise these ratings. The necessary understandability metrics are in the Appendix 2.

6.1.5 Evaluating the Correctness of NN

Correctness refers to the syntactical and grammatical correctness of the model regarding the used modelling language. Further a correct model does not contain redundancies [51]. Correctness can be evaluated easily because there is very little subjectivity involved, and no degrees of quality. The model either follows the modelling language's rules or it does not. Also, the model can be evaluated in isolation, without reference to user requirements [51]. The result of correctness reviews will be a list of defects, defining where the data model does not conform to the rules of the data modelling technique. Many the syntactical and grammatical checks can be carried out automatically using CASE tools.

The proposed quality measures for correctness all take the form of defects with respect to data modelling standards (syntactic rules). We break down correctness errors into different types or defect classes to assist in identifying patterns of errors or problem areas which may be addressed by training or other process measures. The purpose of the review process will be to eliminate all such defects.

6.1.6 Evaluating the Simplicity of NN

Simplicity means that the data model contains the minimum possible constructs [51]. Simpler models are more flexible [56], easier to implement [57], and easier to understand [58]. If there are two data models which meet the same requirement the simpler one should be preferred.

Simplicity can be evaluated easily because it only requires only counting of data model elements. This can be done automatically by CASE tools, or carried out manually. Simplicity metrics are particularly useful in comparing alternative data models [51].

Metrics for evaluating simplicity take the form of complexity measures. The simplicity metrics are given in the Appendix 2.

6.1.7 Evaluating the Integration of NN

Integration is defined as the level of consistency of the data model with the rest of the organisation's data [51]. An approach for achieving corporate-wide data integration is the mentioned corporate data. This data model provides a common set of data definitions which is used to co-ordinate the activities of application development teams so that separately developed systems work together. The corporate data model allows opportunities for sharing of data to be identified, and ensures that different systems use consistent data naming and formats [59].

Integration evaluation is based on comparing data model with the corporate data model. The result of this will be a list of conflicts between the project data model and the corporate data model [51]. This is usually the responsibility of the data administrator (also called information architect, data architect, data manager), who has responsibility for corporate-wide sharing and integration of data. It is their role to maintain the corporate data model and review application data models for conformance to the corporate model.

Most of the metrics for integration consider conflicts with the corporate data model or with existing systems. The purpose of the review process will be to resolve these inconsistencies. In the Appendix 2, the integration metrics are given.

6.1.8 Evaluating Implementability (Feasibility) of NN

Implementability is defined as the ease with which the data model can be implemented within the time, budget and technology constraints of the project [51]. Although it is important that a data model does not contain any implementation relevant information [52] it is also important that it does not ignore all practical considerations. After all, there is little point developing a model which cannot be implemented or that the user cannot afford [51].

The Implementation is assessed by the developers implementing it. The assessment considers feasibility and the relation to the expected costs and time. The process of reviewing the model also allows the application developer to gain familiarity with the model prior to the design stage to ensure a smooth transition [51]. The implementability metrics are introduced in Appendix 2.

As a conclusion, Moody's Framework seems to be quite heavy because it defines 25 metrics. The author admits that not all metrics are necessary for an evaluation but the framework primary aims at being complete covering all quality factors [51]. Criteria for choosing the metrics should be the metric's perceived usefulness and ease of calculation.

7 Systems/Application Architecture - Software Architecture Evaluation Techniques

The Systems/Application Architecture defines the software systems which is necessary to process the data and support the business. The software system is described by the *software architecture*. The software architecture basically must describe the software system's components. That means their structure as well as their behaviour and interaction with each other because the whole software system's behaviour results from its components' behaviour. The authors of [2] define software architecture as follows:

> The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them.

An architecture evaluation can be performed in different stages of architecture creation process. Actually, the authors of [60], [61] distinguish two possible evaluation phases: the *early* and *late evaluation*. Depending on the stage of the architectural outputs different methods can be applied. In the following the two phases and the evaluation methods are presented.

7.1 Early and Late Architecture Evaluation

Early evaluation is performed when only fragments of the architectural description exist so that mostly the techniques questionnaires, checklists, and scenario-based methods are used for assessment because at this stage there is not enough tangible information available for collecting metrics or simulating behaviour. Mainly the experience of the developers and scenarios based on requirements in the requirement documents are the foundation for the early evaluation. The questionnaires and scenario-based techniques have a stronger focus on evaluating whether the stakeholders' requirements are met by the architecture, and the identification and evaluation of the relevant design decisions implementing these requirements.

Late architecture evaluation is carried out during later stages of the software development process when there is at least a detailed design available on which more concrete metrics can be collected. To ensure the quality control and quality assurance early evaluation and late evaluation techniques should be used in this way. It is possible to ensure that the stakeholders' requirements are considered and implemented in the architecture. This point of view corresponds with [62] because the author proposes first an architectural review which is actually an early evaluation and secondly the determination of relevant quality attributes by applying techniques like architectural metrics [60], simulation ([63], [64]) and mathematical modelling ([62], [65]). In fact, the second proposition, given by [62], has the same purpose as the late evaluation. Next, some techniques are briefly described.

7.2 Questionnaires and Checklist

According to [2], the techniques using questionnaires and checklists are quite similar; both consist of questions regarding the issue if the architecture fulfils functional and non-functional requirements. These questions have to be answered by a group of the software system's

stakeholders. That means this evaluation is based on their experience. Questionnaires as well as checklists are assessed statistically. An example of the questionnaire and checklist techniques is presented in Appendix 3.

7.3 Scenario-based Methods

The following explanation is based on the book [60] whose authors developed several scenario-based methods at the Software Engineering Institute at Carnegie Mellon University.

Scenario-based techniques evaluate the software architecture by considering it from a higher abstraction level that means the architectural description must neither be complete nor very detailed. A further commonness is that that these methods define a number of steps which have to be performed to achieve a useful evaluation result. These steps are:

- description of the architecture or the architectures which should be evaluated
- development of scenarios (based on non-functional requirements)
- prioritization of the scenarios according to the quality attributes they should prove
- evaluation the architecture from the high-priority scenarios perspectives
- exposition of the results.

The scenarios describe the desired system's behaviour during performing certain tasks. This behaviour depends on the existence of certain quality characteristics. That means if the architecture enables the fulfilment of certain scenarios proves the implementation of certain quality characteristics. The quality of the evaluation and especially its results depends on the scenarios' quality. Their quality increases by a well done mapping of requirements to scenarios. It is fatal if an important and necessary scenario is missing during the evaluation. Therefore, the scenario development should involve representatives from all stakeholders. Such scenario-based methods are for example:

- Software Architecture Analysis Method (SAAM)
- Architectural Trade-off Analysis Method (ATAM)
- Active Reviews for Intermediate Designs (ARID)
- Architecture-Level Modifiability Analysis (ALMA)
- Family Architecture Assessment Method (FAAM)
- Cost Benefits Analysis Method (CBAM)

The *Cost Benefits Analysis Method* (CBAM) aims at the estimation of factors cost and time related to the benefit of a design decision. So, usually, this method follows after the identification and analysis of design decisions and the trade-offs and risks related to them. In Table 2, a comparison of the above listed methods is presented.

Method	Assessed Quality	Metrics and Tools Support	Process Description	Strengths	Weaknesses	Systems Type Applicable for
SAAM	Modifiability	Scenario classification (direct vs. indirect ones)	Reasonable	Identifying the areas of high potential complexity Open for any architectural description	Not a clear metric Not supported by techniques for performing the steps	All
ATAM	Modifiability	Sensitivity Points, Tradeoff Points Supported by ATA Tool	Good	Scenario generation based on Requirements Applicable for Static and dynamic properties Quality utility tree	Requires detailed technical knowledge	All
CBAM	Costs, Benefits, and Schedule Implications	Time and costs	Reasonable	Provide business measures for particular system changes Make explicit the uncertainty associated with the estimates	Identifying and trading costs and benefits can be done by the participants in an open manner	All
ALMA	Modifiability	Impact estimation, Modifiability prediction Model	Reasonable	Scenario generation stopping criterion	Restricted set of case studies Concentrates on static properties	Business Information Systems
FAAM	Interoperability and Extensibility	Various specialized tables and Diagrams	Very good Detailed process flow	Emphasis on empowering the teams in applying the FAAM session	Only partially proven in one particular environment Concentrates on static properties	System Families

Table 2: Comparison of scenario-based methods [66]

The comparison shows that SAAM, ATAM and CBAM can be applied on any kind of a system. Furthermore, these methods have been applied and validated in several industrial cases. Since ATAM is a successor of SAAM and results in more tangible information regarding risks and trade-offs cause by design decision ATAM will be described in the following. Also the CBAM evaluation is described afterwards because it enables time, cost and benefits analysis.

7.3.1 Architectural Trade-off Analysis Method (ATAM)

The ATAM is so named because it reveals how well the architecture satisfies particular quality goals and since it recognizes that architectural decisions affect more than one quality attribute that means this method enables the identification of trade-offs among several quality attributes. According to [60] the participation of three different groups is usually necessary for performing the ATAM. The groups are the evaluation team, the project decision makers and the architecture stakeholders. The groups and especially the evaluation team are described in more detail in the Appendix 4.

The whole ATAM-based evaluation is divided into four phases. The first phase is called *partnership and preparation*. In this phase basically the evaluation team leadership and the key project decision makers informally meet to work out the details of planned evaluation. They agree on formal issues like logistics, such as the time and place of meetings, statement of work or nondisclosure agreements, and then they agree about a preliminary list of stakeholders.

Furthermore, they decide which architectural documents will be delivered to the evaluation team for performing the evaluation. The actual *evaluation phases* are the second and third phase. The evaluation team uses the second phase for studying the architecture documentation to get a concrete idea of what the system is about, the overall architectural approaches which are chosen, and the

important quality attributes. During the third phase the system's stakeholders join the evaluation team and both groups analyze the architecture together. The analysis is based on the elicitation of scenarios. According to [60] the capturing and elicitation of functional and non-functional requirements is part of ATAM.

In the fourth and last phase the evaluation team creates and delivers the final report. The concrete steps which are performed during the ATAM evaluation are described in the Appendix 5.

The ATAM evaluation results in the following outputs:

- architectural approaches documented
- set of scenarios and their prioritization from the brainstorming
- utility tree
- risks
- non-risks
- sensitivity points and trade-off points

7.3.2 Cost-Benefit Analysis Method (CBAM)

CBAM begins where ATAM leaves off because this method enables analyzing the costs, benefits and schedule implications of architectural decisions. Different form the former method CBAM is bridging two domains in software development the architecting process and the economics of the organization. CBAM is adding the costs (and implicit budgets or money) as quality attributes, which need to be considered among the trade-offs when a software system is going to be planned. ATAM (and SAAM) primarily considered the design decisions with respect to architectural quality attributes like modifiability, performance, availability, usability, and so on. CBAM is focusing on costs, benefits and risks which are as important as the other quality attributes and they are relevant to be considered when the architectural decisions are being made. The impulse of the CBAM development came from a set of questions, each of which contributed in shaping the method. These questions were addressed as:

- How can the architectural decisions be measured and compared in terms of their different implications, costs and benefits?
- How can quality attributes be analyzed and trade-off with respects to their costs and benefits involved?
- How can be characterized the uncertainty level associated with these cost and benefits estimates?

The evaluation team (Appendix 6) in accordance with the project scale and goals must appreciate the effort. Looking at the organizational aspects and CBAM steps (Appendix 6) one can say that most of the effort is concentrated in architectural strategies elicitation and cost-benefit-schedule prediction part. A CBAM session takes one or two days. In addition an ATAM allocated is increasing to at least four working days. In terms of man-hours estimation and procedural costs, the CBAM team can provide certain effort values.

CBAM general strengths and outputs are:

- values as a basis for a rational decision making process in applying certain architectural strategies
- business measure that can determine the level of return on investment of a particular change to the system
- help for organizations in analyzing and pre-evaluating the resource investment in different directions by adopting those architectural strategies that are maximizing the gains and minimize the risks
- Since CBAM is built on the general architecture assessment methods like SAAM and ATAM, the method is inheriting their benefits with respect to efficiency.

So far, there is no method that incorporates the economical perspective in the software quality attributes evaluation and trade-off analysis.

7.4 Architectural Metrics

This approach aims at measuring certain attributes of the software architecture which enable assumptions about the architecture's quality. Architectural metrics belong to the group of product metrics as described in [67]. They are derived from quality attributes which are refined quality characteristics. The existing software architectural metrics are quite limited. Furthermore, the so-called architectural metrics are very similar to design metrics. A reason for this is, according to [60], that the existence of a detailed architectural description is necessary to collect metrics. That means that the design description is at a stage where it can be implemented or parts of it are already implemented. Mostly metrics about structural characteristics are collected. These measurements are performed with the help of the architectural descriptions which are commonly presented in UML notation or on the code level; this enables partly tool-based measurements. The architectural metrics reflect class characteristics like the complexity of a class, number of methods, depth of the inheritance hierarchy, coupling, and cohesion. The collected metrics are interpreted for evaluating quality attributes, especially maintainability, testability, understandability, reusability, complexity, and also efficiency. The three common metrics cohesion, coupling, and Cyclomatic Complexity are described in the Appendix 7.

7.5 Prototyping

Prototyping has been described e.g. in [63], [64]. In prototyping, the most important quality attributes are refined into scenarios. The essential functionality to perform these scenarios is implemented in the prototype. The executable prototype can be tested regarding quality attributes at runtime. The gained results are used for further development or correction of the software architecture. The scenarios are mostly implemented without user-oriented and business-oriented aspects of the architecture, what makes the prototyping evaluation approach resource-saving especially regarding time and cost. The prototyping approach is often also called simulation in the literature, e.g. in [62].

7.6 Mathematical Modelling

A mathematical model is an abstract model which describes the system's behaviour or certain aspects of the system's behaviour. The model is used for determining theoretically how the system reacts on certain events. According to ([62], [65]), especially for high-performance computing,

reliable systems, real-time systems, etc. mathematical models have been developed, and they can be used to evaluate especially quality attributes related to the runtime behaviour of the system. Different from the other approaches, the mathematical models allow for static evaluation of architectural design models. Mathematical modelling is an alternative to prototyping because both approaches are primarily suitable for assessing runtime behaviour. The approaches can also be combined. Two widely spread types of models are *performance modelling* and *real-time task models*. For example, performance modelling can be used to determine the computational requirements of the individual components in the architecture. These theoretical results can then be used and proofed with the running prototype in a simulation. Since the focus of this work also is on the performance assessment with the help of the architecture performance modelling is a suitable approach. Typical performance models are queuing networks and Markov chains which are based on stochastic and probability-based methods, and other stochastic approaches like stochastic process algebras.

7.7 Summary

While the measurement-based approaches, architectural metrics, and prototyping give concrete values for the evaluation and make it that way a bit sounder, they have the drawback that they can be applied only in the presence of a working artefact. Also the mathematical models are based on detailed description of the whole architecture or at least of some components because the more detailed the model the more realistic are the computed results. Questionnaires and scenario-based evaluations, on the other hand, work just fine on hypothetical architectures, and can be applied much earlier in the life cycle. Actually, these techniques can be seen as architectural review with the main stakeholders because they improve the understanding of the impact of architectural decisions on the system's requirements. Furthermore, even if the architectural description is not in the implemental stage, these approaches are able to identify insufficiencies, weaknesses, and risks. Especially the utilization of ATAM and CBAM is a promising approach to evaluate a software system’s quality and costs.

8 Technology Architecture Evaluation

The Technology Architecture describes the hardware and communication technology which is used within the organization to enable the communication and to deploy the utilized software [68]. Technology architecture includes [11]:

1. Hardware and platforms
2. Local and wide area networks
3. Operating System
4. Infrastructure software such as application servers, database management system and middleware.

The runtime behaviour of the software system supporting the organization's processes is strongly depending on the underlying technology therefore the planning and design of a software system should already consider the underlying platforms. Common software architecture models like Kruchten's *4+1 views* [69] or Soni's model [70] include also a description of the execution environment. Hence, the technology can be evaluated as part of the software system during the software architecture evaluation. Usually, the components used within the technology architecture are commercial-of-the-shelf (COTS) components and their quality characteristics are described by the supplier. However, it is necessary to integrate different components with each other and different implementations have different behaviour concerning runtime characteristics. Therefore the infrastructure can be evaluated by using benchmarking. Benchmarking primary evaluates performance, scalability and reliability of the used infrastructure. The evaluation results gained from benchmarking can be compared to the expected costs which are connected to different COTS components. That cost/benefit consideration supports decision making regarding the questions which COTS components suit best the organization's software systems.

9 Mapping of Methods to Architecture Evaluation Needs

In this section, the presented evaluation methods are assigned to the evaluation needs mentioned in Section 1.

Table 3 shows the mapping of evaluation needs to the presented evaluation methods. The methods which are mapped to the needs are suggestions for assessing the needs and concerns relating to enterprise and software architecture. Furthermore, the satisfaction of the needs is strongly depending on the used input for the evaluation, especially, the architectural artefacts and the skills and experience of the evaluation teams. It should be noticed, that Table 3 takes into account only those needs for which it was possible to find evaluation methods.

The suggested methods evaluate the architecture regarding concerns related to the demanded evaluation needs. However, it is difficult to predict the extent of satisfaction for certain needs because the needs definitions are rather general. Only the application of the methods to the specific EA can answer the question how well the suggested methods satisfy the evaluation needs of a specific organization. Furthermore, the combination of methods might be necessary to improve the fulfilment of certain needs.

Table 3 Mapping of Evaluation needs to methods

Method Name	Technique	Strengths	Practical Proof	Means of Implementation	Addressed Evaluation Needs
Business Architecture					
Governance Modelling	conceptual modelling and review	• vision, goals, objectives are made explicit • transparency of transformation drivers • tracing of decisions and responsibilities • basis for analysis and evaluation (conflicts, improvement, level of fulfilment) • basis for planning and changing strategies and processes	standardized by OMG	Business Motivation Model (BMM)	• observation that ICT-architecture do not correspond to company's business's requirements • enhances the understanding of company's business/ICT • enhances the understanding of responsibilities in the company • make sure that organisational choices are suitable • An effort towards long-term technical solutions and need to argue for the long-term technical solutions
Business Process Modelling	conceptual modelling and review	• visualization of processes regarding relationships, dependencies, and effects between processes and their activities and resources • visualization increases the understanding about the processes and supports the validation	yes	BPMN, EPC, ARIS and many other tools	• change need in the business or ICT (e.g. a need to move from one solution to another) • observation that ICT-architecture do not correspond to company's business's requirements • enhances the understanding of company's business/ICT • enhances the understanding of responsibilities in the company • make sure that organisational choices

Method Name	Technique	Strengths	Practical Proof	Means of Implementation	Addressed Evaluation Needs
		and improvement for many stakeholders • 80% of process advancements are achieved by modelling the current status			are suitable • distribution of work • Business process planning • need to find the best possible system solution and a need to understand the aspects relating the solution • An effort towards long-term technical solutions and need to argue for the long-term technical solutions
Business Process Simulation	simulation	• the current processes (*as-is* state) regarding costs, performance • analyze *what-if* scenarios, obtain cost and performance predictions, and validate processes • support the decision making regarding organizational change and future investments	yes	ARIS Simulation, BPEL	• change need in the business or ICT (e.g. a need to move from one solution to another) • observation that ICT-architecture do not correspond to company's business's requirements • make sure that organisational choices are suitable • Business process planning • need to find the best possible system solution and a need to understand the aspects relating the solution • An effort towards long-term technical solutions and need to argue for the long-term technical solutions
Business Value	priority-based	• supports the	by Intel		• change need in the business or ICT

Method Name	Technique	Strengths	Practical Proof	Means of Implementation	Addressed Evaluation Needs
Index (BVI)	assessment of future investments	prioritization of investment options • tangible and intangible value can be measured			(e.g. a need to move from one solution to another) • effort to drive investments to follow up architectural principles
Total Economic Impact (TEI)	Risk-adjusted Return on Invest calculation	• measures cost, benefits, flexibility, and risk impact on business • risk-adjusted ROI	by Forrester	Forrester's implementation support	• change need in the business or ICT (e.g. a need to move from one solution to another) • understanding quality aspects relating to the company's application portfolio • effort to drive investments to follow up architectural principles
ValIT	Value governance, Portfolio management, and investment management	• Value governance • Portfolio management • Investment management	validated by the Dutch financial services firm ING	support from IT Governance Institute (ITGI)	• change need in the business or ICT (e.g. a need to move from one solution to another) • understanding quality aspects relating to the company's application portfolio • effort to drive investments to follow up architectural principles
Applied Information Economics (AIE)	IT investment assessment through mathematical and scientific methods	• mathematical models • Developing financially-based quality assurance measures • Developing a strategic plan for information systems	not wide-spread because the method is very complex		• change need in the business or ICT (e.g. a need to move from one solution to another) • effort to drive investments to follow up architectural principles

Method Name	Technique	Strengths	Practical Proof	Means of Implementation	Addressed Evaluation Needs
Information Architecture					
Moody's Framework	reviews and metrics	• evaluates data model's quality • provides quantitative measures • coverage of many data model quality aspects	yes	Entity-Relationship modelling, Moody's Framework	• information / data models of good quality • understanding information managed in company
Software Systems Architecture					
SAAM	scenario-based review aims on scenario validation	• knowledge transfer about architectural decisions • identification of areas of high potential complexity		evaluation steps of the software engineering institute, Carnegie Mellon	• understanding the state of the company's application portfolio • understand the current state of technical infrastructure • need to find the best possible system solution and a need to understand the aspects relating the solution
ATAM	scenario-based review regarding system's quality characteristics including scenario validation, trade-off and risk identification	• identifies risks and points of trade-off • enables evaluation of structural and behavioural system characteristics • improves architectural knowledge sharing	yes	evaluation steps of the software engineering institute, Carnegie Mellon	• change need in the business or ICT (e.g. a need to move from one solution to another) • need to enhance the understanding of company's business/ICT • understanding the state of the company's application portfolio • understanding quality aspects relating to the company's application portfolio • understanding the current state of technical infrastructure

Method Name	Technique	Strengths	Practical Proof	Means of Implementation	Addressed Evaluation Needs
					• need to find the best possible system solution and a need to understand the aspects relating the solution • An effort towards long-term technical solutions and need to argue for the long-term technical solutions
CBAM	scenario-based review with focus on cost and benefits	• measurement of design decisions with cost and benefit metric • makes uncertainty explicit associated with the estimates	yes		• change need in the business or ICT (e.g. a need to move from one solution to another) • understanding quality aspects relating to the company's application portfolio • effort to drive investments to follow up architectural principles • An effort towards long-term technical solutions and need to argue for the long-term technical solutions
Technology/Infrastructure Architecture					
Benchmarking	Measures performance, reliability, and cost	• enables the collection of metrics regarding the system's performance, reliability and cost • supports decision making	yes	Benchmark test tools	• understanding the current state of technical infrastructure

10 Conclusions

This paper dealt with the topic of architecture evaluation methodologies, especially focusing on methods for enterprise and software architecture evaluation. While there are several methods for evaluating architectural artefacts in the area of software architectures there seems to be a lack of methodologies evaluating enterprise architecture. The most wide-spread approaches are maturity models and IT-Business-Alignment assessment methods. However, they address primarily the enterprise architecture management and development process and not the evaluation of architectural outputs.

Most of the architecture evaluation needs described in this report (Section 2) refer to concerns which have to be assessed through analysis of architectural descriptions. Since enterprise architecture is a composition of different architectural views addressing different concerns, this paper suggested means to assess these views regarding the detected needs. Methodologies to evaluate the business, information, systems and technology architectures were presented. Many methods rely on conceptual modelling to be understandable for different stakeholders from different domains such as managers, business analysts, and developers. Therefore, conceptual modelling standards, such as BPMN, which enhances the understanding, knowledge sharing and the analysis of the structure and behaviour of the organization, are considered as evaluation approaches. The evaluation techniques suggested in this paper are a collection of review methods analyzing conceptual models, simulation approaches, and measures for predictions relating to the changing environment but also metrics for assessing quality attributes. All presented assessment techniques are either based on standards or are developed or validated in a practical environment. In the following, the suggested approaches are briefly summarized.

For evaluating the business architecture the following methods were presented:

- governance modelling (improvement of tracing between vision/goals and processes and tasks)
- business process modelling and simulation (enhancing knowledge and enabling what-if-scenarios)
- financial methods for assessing the value of IT investment (prediction of expected benefits through IT investment)

The needs concerning the enterprise's information architecture were addressed by evaluation of the corporate data model which is a structured conceptual model of the organisation's data entities and their relations. The suggested methodology was *Moody's Framework*.

The systems architecture consists of software systems. A software system is described through software architectural artefacts. Therefore, the evaluation techniques suggested for the systems architecture are methods for software architecture evaluation. Since the infrastructure which allows the deployment of software applications is also part of the software system, the underlying execution environment can be evaluated within the software architecture evaluation. The methods concerning the software system evaluation enable predictions regarding the whole system life cycle. Especially, characteristics, such as performance, cost, reliability and maintenance are essential

characteristics in the enterprise architecture context. The suggested methods are able to assess these criteria.

A further conclusion is the fact that architecture evaluation depends strongly on conceptual models which are used to share and communicate the architectural knowledge among different stakeholders from different domains. Therefore, conceptual modelling standards are part of the evaluation methods or conceptual models belong to the evaluation input and are the basis for analysis and discussion about architectural decisions.

This report showed that there are techniques to evaluate enterprise architecture with the help of architectural descriptions. However, the complexity of enterprise architecture and the related variety of concerns complicates reaching an established overall evaluation approach. The problem of developing methodologies enabling the enterprise architecture evaluation in a coherent, efficient, and practical way should be overcome in future research and work.
So far it is only possible to apply different techniques on only single architectural views of EA. Integrating these techniques into the EA evaluation process of a company might be difficult. These techniques are independent of each other and they refer to different standards, description models, and tools which are not compatible to those already used within in the organization.

11 References

1. Niina Hämäläinen, T.Y., Eetu Niemi, *The Role of Architecture Evaluations in ICT-companies*. 2007, Information Technology Research Institute, University of Jyväskylä.
2. Bass, L., P. Clements, and R. Kazman, *Software Architecture in Practice*. 2003: Addison-Wesley. 560.
3. Wiegers., K.E., *Software Requirements 2: Practical techniques for gathering and managing requirements throughout the product development cycle*. 2 ed. 2003: Microsoft Press.
4. Heinen, E., *Grundfragen der entscheidungsorientierten Betriebswirtschaftslehre*. 1972, München.
5. Hilliard, R., M.J. Kurland, and S.D. Litvintchouk, *MITRE's Architecture Quality Assessment*, in *Proceedings of the Software Engineering & Economics Conference*. 1997.
6. Hämäläinen, N., J. Ahonen, and T. Kärkkäinen. *Why to Evaluate Enterprise and Software Architectures - Objectives and Use Cases*. in *Proceeding of the 12th European Conference on Information Technology Evaluation*. 2005. Turku.
7. Lopez, M., *An Evaluation Theory Perspective of the Architecture Tradeoff Analysis Method (ATAM)*. 2000, The Software Engineering Institute, Carnegie Mellon University: Pittsburg, USA.
8. Ylimäki, E.N.a.T., *Enterprise Architecture Evaluation Components*, in *Abstract accepted to HAAMAHA 2007*. 2007.
9. ISO, *ISO/IEC 9126-1:2001, Software engineering -- Product quality -- Part 1: Quality model*. 2001, ISO.
10. F. Lasavio, L.C., *ISO quality standards for measuring architectures.* The Journal of Systems and Software, 2004. **72**: p. 209 - 223.
11. Zachman, J.A., *A Framework for Information Systems Architecture.* IBM Systems Journal, 1987. **26**(3): p. 276-292.
12. The Open Group. *The Open Group Architecture Framework version 8.1.1, Enterprise Edition (TOGAF 8.1.1).* 2006 [cited 2006 10 September]; Available from: http://www.opengroup.org/architecture/togaf/.
13. CIO Council, *Federal Enterprise Architecture Framework, Version 1.1., September 1999*. 1999, The Chief Information Officers Council (CIO).
14. Defense, D.o., *Department of Defense Architecture Framework Version 1.0 - Vol 1 Defintion & Guideline and Vol 2 Product Descriptions.* 2003.
15. Paulk, M.C., Curtis, B, Chrissis, M.B., and Weber, C.V, *Capability Maturity Model, Version 1.1.* IEEE Software, 1993. **10**: p. 18-27.
16. Gartner, *Return on Enterprise Architecture: Measure It in Asset Productivity*, in *GartnerG2 Report*. 2002, Gartner, Inc: Stamford, USA.
17. META Group Inc., *Architecture Capability Assessment.* META Practice, 2000. **4**(7).
18. OMB. *Guidelines for Enterprise Architecture Assessment Framework.* 2004 [cited.
19. DoC, *IT Architecture Capability Maturity Model*, Department of Commerce (DoC), Editor. 2003.
20. NASCIO, *NASCIO Enterprise Architecture Maturity Model, v. 1.3*. 2003, National Association of State Chief Information Officers (NASCIO).
21. Luftman, J., *Assessing Business-IT Alignment Maturity.* Communications of the Association for Information Systems, 2000. **4**(Article 14).

22. Chan, Y.E., *Why Haven't We Mastered Alignment? The Importance of tthe Informal Organizational Structure.* MIS Quarterly Executive, 2002. **1**(2).
23. CIO Council, *Architecture Alignment and Assessment Guide*. 2000, Chief Information Officers Council.
24. E. Yu, M.S., X. Deng. *Exploring Intentional Modeling and Analysis for Enterprise Architecture*. in *10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06)*. 2006: IEEE.
25. The Business Rules Group (BRG), *Business Motivation Model (BMM) Release 1.2*. 2005, The Business Rules Group.
26. Object Management Group (OMG), *Business Motivation Model (BMM) Specification*. 2006, OMG.
27. J. Mylopoulos, L.C., E. YU, *From object-oriented to goal-oriented requirements analysis.* Communications of the ACM, 1999. **42**(1): p. 31 - 37.
28. Yu, E. *Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering*. in *the 3rd IEEE International Symposium on Requirements Engineering (RE'97)*. 1997: IEEE Computer Society.
29. Jørgensen, J.K.a.H., *Interactive Models for Supporting Networked Organisations*, in *Advanced Information Systems Engineering*. 2004, Springer Berlin / Heidelberg.
30. D. I. Vidovic, V.B.V. *Dynamic business process modelling using ARIS*. in *Information Technology Interfaces*. 2003. Cavtat, Croatia.
31. Ali Bahrami, D.S., Soheila Bahrami. *Enterprise Architecture for Business Process Simulation*. in *Simulation Conference Proceedings*. 1998. Winter.
32. Scheer, A.-W. (1996) *ARIS-House of Business Engineering*. IWi Hefte **Volume**, DOI: 133
33. Chen, P., *The Entity-Relational model: Towards a unified view of data.* 1976.
34. V. Stefanov, B.L. *A Performance Measurement Perspective for Event-Driven Process Chains*. in *16th International Workshop on Database and Expert Systems Apllications (DEXA'05)*. 2005: IEEE.
35. OMG, *Business Process Modeling Notation Specification*. 2004, Object Management Group (OMG).
36. A. Arkin, S.A., B. Bloch, F. Curbera, Y. Goland, N. Kartha, C. Liu, S. Thatte, P. Yendluri, and A. Yiu, editors., *Web Services Business Process Execution Language Version 2.0. Working Draft*. 2005, WS-BPEL TC OASIS.
37. Ouyang Chun, M.D., A.H.M ter Hofstede, W.M.P. van der Aalst. *From BPMN Process Models to BPEL Web Services*. in *IEEE International Conference on Web Services (ICWS'06)*. 2006: IEEE.
38. Symons, C., *Measuring The Business Value Of IT - A Survey Of IT Value Methodologies.* 2006.
39. Intel, *Managing IT Investments - Intel's IT Business Value metrics program*. 2003, Intel Technology.
40. ITGI, *ENTERPRISEVALUE: GOVERNANCE OF IT INVESTMENTS - The Val IT Framework*. 2006, IT Governance Institute.
41. ITGI, *ENTERPRISEVALUE: GOVERNANCE OF IT INVESTMENTS - The ING Case Study*. 2006.
42. IT Governance Institute, *Governance of the Extended Enterprise: Bridging Business and IT Strategies*. 2005, Hoboken, USA: John Wiley & Sons.
43. Federal Electronics Challenge, *Applied Information Economics Analysis for Desktop Replacement (EPA)*. 2006.

44. Halttunen, V., *Architectural Planning of Information Systems: A Structure for Coping with Diversified Architectures.*, in *Larkki project report, 8.2.2002*. 2002.
45. G. A. Lewis, S.C.-D., P. Place, D. Plakosh, R. C. Seacord, *An Enterprise Information System Data Architecture Guide*. 2001, Software Engineering Institute Carnegie Mellon (SEI).
46. D. L. Goodhue, L.J.K., M. D. Wybo, *The Impact of Data Integration on the Costs and Benefits of Information Systems.* MIS Quarterly, 1992. **16**(3): p. 293-311.
47. O. Lindland, A.S., A. Solvberg, *Understanding Quality in Conceptual Modeling.* IEEE Software, 1994. **11**: p. 42 - 49.
48. D. L. Moody, G.G.S. *What Makes a Good Data Model? Evaluating the Quality of Entity Relationship Models*. in *the Thirteenth International Conference on the Entity Relationship Approach*. 1994. Manchester: Springer, Berlin.
49. Kesh, S., *Evaluating the Quality of Entity Relationship Models.* Information and Software Technology, 1995. **37**(12): p. 681 - 689.
50. R. Schuette, T.R. *The Guidelines of Modeling - An Approach to Enhance the Quality in Information Models*. in *Seventeenth International Conference on Conceptual Modelling (ER'98)*. 1998. Singapore: Springer-Verlag.
51. D.L. Moody, G.G.S., P. Darke. *Improving the Quality of Entity Relationship Models - Experience in Research and Practice*. in *Seventeenth International Conference on Conceptual Modelling (ER '98)*. 1998. Singapore: Springer-Verlag.
52. ISO, *Information Processing Systems - Concepts and Terminology for the Conceptual Schema and the Information Base*. 1987, ISO.
53. C. Batini, S.C., S. B. Navathe, *Conceptual Database Design: An Entity Relationship Approach*. 1992, Redwood City, California: Benjamin Cummings.
54. Date, C.J., *Introduction to Database Systems*. 4 ed. 1989: Addison Wesley.
55. R. S. Loffman, R.M.R., *Improving Data Quality.* Database Programming and Design, 1991. **4**(4): p. 17 - 19.
56. Meyer, B., *Object Oriented Software Construction*. 1988, New York: Prentice Hall.
57. Simsion, G.C. *Creative Data Modelling*. in *Tenth InternationalEntity Relationship Conference*. 1991. San Fransico.
58. Moody, D.L. *A Multi-Level Architecture for Representing Enterprise Data Models*. in *Sixteenth International Conference on the Entity Relationship Approach*. 1997. Los Angeles.
59. Martin, J., *Strategic Data Planning Methodologies*. 1989, New Jersey: Prentice Hall.
60. Clements, P., R. Kazman, and M. Klein, *Evaluating Software Architectures: Methods and Case Studies*. 1st Edition ed. 2001, Boston, USA: Addison-Wesley.
61. Svahnberg, M., *An Industrial Study on Building Consensus Around Software Architectures and Quality Attributes.* Information and Software Technology, 2004. **46**: p. 805-818.
62. Bosch, J., *Software architecture assessment*, in *International Summer School on Usability-Driven Software Architecture*. 2005, University of Technology: Tampere, Finland.
63. A. V. Corry, J.B., H. B. Christensen, M. Ingstrup, and K. M. Hansen. *Exploring quality attributes using architectural prototyping*. in *Proceedings of the First International Conference on the Quality of Software Architectures (QoSA 2005)*. 2005: Springer-Verlag, Berlin Heidelberg.
64. H. Grahn, M.M., F. Mårtensson. *An approach for performance evaluation of software architectures using prototyping*. in *7th IASTED International Conference on Software Engineering and Applications*. 2003.

65. Bosch, J. and P. Molin, *Software Architecture Design: Evaluation and Transformation*, in *Proceedings of the IEEE Conference and Workshop on Engineering of Computer-Based Systems, ECBS '99*. 1999, IEEE Computer Society: Nashville, TN, USA. p. 4-10.
66. Ionita, M.T., D.K. Hammer, and H. Obbink, *Scenario-Based Software Architecture Evaluation Methods: An Overview*, in *Proceedings of the International Conference on Software Engineering (ICSE 2002)*. 2002: Orlando, Florida, USA.
67. IEEE, *IEEE Standard for a Software Quality Metrics Methodology, IEEE Std 1061-1998*. 1998.
68. Schekkerman, J., *How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework*. 2003, Trafford Publishing.
69. Kruchten, P., *4+1 View Model of Architecture*. IEEE Software, 1995. **12**(6): p. 42-50.
70. Soni, D., R.L. Nord, and C. Hofmeister. *Software architecture in industrial applications*. in *Proceedings of the 17th international conference on Software engineering*. 1995. Seattle, Washington, United States: ACM Press.
71. Klir, G.J., *Architecture of Systems Problem Solving*. 1985, New York: Plenum Press.
72. Pippenger, N., *Complexity Theory*. Scientific America, 1978. **238**: p. 1 - 15.
73. Thompson, C., *"Living with an Enterprise Model"*. Database Programming and Design, 1993. **6**: p. 32 - 38.
74. Barbacci, M., et al., *Quality Attributes, Technical Report CMU/SEI-95-TR-021*. 1995, Software Engineering Institute, Carnegie Mellon University.
75. Lindvall, M., R. Tesoriero, and P. Costa, *Avoiding Architectural Degeneration: An Evaluation Process for Software Architecture*, in *Proceedings of the Eighth IEEE Symposium on Software Metrics (METRICS'02)*. 2002, IEEE Computer Society. p. 77-86.
76. McCabe, T.J., *A complexity measurement*. IEEE Transactions on Software Engineering, 1976: p. 308 - 320.

Appendix 1. Business Value Index Example

All three dimensions use a predetermined set of criteria including customer need, business and technical risks, strategic fit, revenue potential, level of required investment, the amount of innovation and learning generated, and other factors [39]. Each dimensions' criteria are weighted according to the ongoing business strategy and its importance to the business environment. Changes in business strategy could change the impact of criteria on a certain dimension.

As project managers or program owners evaluate their proposed investments using the BVI tool, they score their project against these criteria on a scale of 0 to 3, depending how the IT investment will likely perform against a range of values set for a particular assessment criteria. The assessment of criteria belonging to the dimension of Business Value is shown in Table 4. Afterwards the single values of the criteria are summed up to a value representing the dimension.

Since every project or program is represented through the three dimensional vector projects can be compared ranked according to their benefits. The comparisons between projects regarding one to three dimensions are possible. Figure 4 illustrates the ranking concerning all three dimensions.

Table 4 Sample Assessment Criteria and Scoring [39]

Criteria	Weight	0	1	2	3
Customer pull/need	4	Low	Medium	High	Very high
Customer product cost reduction	3	Increase	No impact	Marginal reduction	Substantial reduction
Business strategic fit and impact	3	Low/NA	Medium	High	Very High
Customer performance improvement	3	Decrease	< 5 %	> 5 %	> 10 %

Appendix 2. Metrics for Data Model Quality

Completeness Metrics:

Metric 1: Number of items in the data model that do not correspond to user requirements. Inclusion of such items will lead to unnecessary development effort and added cost.

Metric 2: Number of user requirements which are not represented in the data model. These represent missing requirements, and will need to be added later in the development lifecycle, leading to increased costs, or if they go undetected, will result in users not being satisfied with the system

Metric 3: Number of items in the data model that correspond to user requirements but are inaccurately defined. Such items will need to be changed later on the development lifecycle, leading to rework and added cost, or if they go undetected, will result in users being unsatisfied with the system.

Metric 4: Number of inconsistencies with process model. A critical task in verifying the completeness of the data model is to map it against the business processes which the system needs to support. This ensures that all functional requirements can be met by the model. The result of this analysis can be presented in the form of a CRUD (Create, Read, Update, Delete) matrix. Analysis of the CRUD matrix can be used to identify gaps in the data model as well as to eliminate unnecessary data from the model [59].

Integrity Metrics:

Metric 5: Number of business rules which are not enforced by the data model. Non-enforcement of these rules will result in data integrity problems and/or operational errors.

Metric 6: Number of integrity constraints included in the data model that do not accurately correspond to business policies (i.e. which are false). Incorrect integrity constraints may be further classified as:

- too *weak*: the rule allows invalid data to be stored
- too *strong*: the rule does not allow valid data to be stored and will lead to constraints on business operations and the need for user "workarounds".

Flexibility Metrics:

Metric 7: Number of elements in the model which are subject to change in the future. This includes changes in definitions or business rules as a result of business or regulatory change.

Metric 8: Estimated cost of changes. For each possible change, the probability of change occurring and the estimated cost for changes made after the implementation should be used to calculate the probability-adjusted cost of the change.

Metric 9: Strategic importance of changes. For each possible change, the strategic impact of the change should be defined, expressed as a rating by business users of the need to respond quickly to the change.

Understandability Metrics:

Metric 10: User rating of understandability of model. User ratings of understandability will be largely based on the concepts, names and definitions used, as well as how the model is presented.

Metric 11: Ability of users to interpret the model correctly. This can be assessed by getting users to instantiate the model using actual business scenarios. Their level of understanding can then be assessed by the number of errors in populating the model. This is a better operational test of understanding than the previous Metric 10 because it measures whether the model is actually understood rather than whether it is understandable [47]. This is much more important from the point of view of verifying the accuracy of the model.

Metric 12: Application developer rating of understandability.

Correctness Metrics:

Metric 13: Number of violations to data modelling conventions. These can be further refined into the following defect classes:

- Diagramming standards violations (e.g. relationships not named)
- Naming standards violations (e.g. use of plural nouns as entity names)
- Invalid primary keys (non unique, incomplete or non-singular)
- Invalid use of constructs (e.g. entities without attributes, overlapping subtypes, many to many relationships)
- Incomplete definition of constructs (e.g. data type and format not defined for an attribute; missing or inadequate entity definition)

Metric 14: Number of normal form violations. Second and higher normal form violations identify redundancy among attributes within an entity (*intra-entity redundancy*). Normal form violations may be further classified into:

- First normal form (1NF) violations
- Second normal form (2NF) violations
- Third normal form (3NF) violations
- Higher normal form (4NF+) violations

Metric 15: Number of instances of redundancy between entities, for example, where two entity definitions overlap or where redundant relationships are included. This is called *inter-entity redundancy*, to distinguish this from redundancy within an entity (*intra-entity redundancy*-Metric14) and redundancy of data with other systems (*external redundancy*-Metric 21)

Simplicity Metrics

Metric 17 is recommended as the most useful of the measures proposed.

Metric 16: Number of entities (E)

Metric 17: Number of entities and relationships (E+R). This is a finer resolution complexity measure which is calculated as the number of entities (E) plus the number of relationships (R) in the data model. This derives from complexity theory, which asserts that the complexity of any system is defined by the number of components in the system and the number of relationships between them ([71], [72]).

Metric 18: Number of constructs (E+R+A). This is the finest resolution complexity measure, and includes the number of attributes in the calculation of data model complexity. Such a metric could be calculated as a weighted sum of the form aNE + bNR + cNA where NE is the number of entities, NR is the number of relationships and NA is the number of attributes. In practice however, such a measure does not provide any better information than Metric 17.

Integration Metrics:

Metric 19: Number of data conflicts with the Corporate Data Model. These can be further classified into:

- Entity conflicts: number of entities whose definitions are inconsistent with the definition entities in the corporate data model.
- Data element conflicts: number of attributes with different definitions or domains to corresponding attributes defined in the corporate data model.
- Naming conflicts: number of entities or attributes with the same business meaning but different names to concepts in the corporate data model. Also entities or attributes with the same name but different meaning to concepts in the corporate data model.

Metric 20: Number of data conflicts with existing systems. These can be further classified into:

- Number of data elements whose definitions conflict with those in existing systems e.g. different data formats or definitions. Inconsistent data item definitions will lead to interface problems, the need for data translation and difficulties comparing and consolidating data across systems.
- Number of key conflicts with existing systems or other projects. Key conflicts occur when different identifiers are assigned to the same object (e.g. a particular customer) by different systems. This leads to fragmentation of data across systems and the inability to link or consolidate data about a particular entity across systems.
- Number of naming conflicts with other systems.
- These are less of a problem in practice than other data conflicts, but are a frequent source of confusion in system maintenance and interpretation of data.

Metric 21: Number of data elements which duplicate data elements stored in existing systems or other projects. This is called *external redundancy* to distinguish it from redundancy within the model itself (Metrics 14 and 15). This form of redundancy is a serious problem in most organizations [51].

Metric 22: Rating by representatives of other business areas as to whether the data has been defined in a way which meets corporate needs rather than the requirements of the application being developed. Because all data is potentially shareable, all views of the data should be considered when the data is first defined [73]. In practice, this can be done by a high level committee which reviews all application development projects for data sharing, consistency and integration.

Implementability Metrics:

The measures of implementability are ratings by the developer:

Metric 23: Technical risk rating: estimate of the probability that the system can meet performance requirements based on the proposed data model and the technological platform (particularly the target DBMS) being used.

Metric 24: Schedule risk rating: estimate of the probability that the system can be implemented on time, based on the proposed data model.

Metric 25: Development cost estimate: this is an estimate of the development cost of the system, based on the data model. Such an estimate will necessarily be approximate but will be useful as a guide for making cost/quality trade-offs between different models proposed. If the quote is too high (exceeds available budget), the model may need to be simplified, reduced in scope or the budget increased.

Appendix 3. Questionnaire and Checklist Example

An example of a questionnaire-based software architecture evaluation is presented in Svahnberg's paper [61]. In this example, the questionnaire used for this evaluation basically aims on the identified necessary system's quality characteristics. According to these quality characteristics, five quality attributes are investigated on four candidate architectures.

The questionnaire contains four parts. The first part covers generic questions like what architecture (e.g. client-server, multi-tier) the participant would prefer based on his/her experience. Moreover, it contains some questions whether there are any architecture types or quality attributes missing. The second part deals with questions to obtain a prioritized list of quality attributes. The third part consists of questions to rate the support given for the quality attributes within each architecture candidate. The fourth part encloses questions to rate which architecture is best at each quality attribute.

Appendix 4. Architecture Trade-Off Analysis Method (ATAM) Participants

Evaluation team is a group of three to five people who are external to the project whose architecture is being evaluated. Each member of the team is assigned a number of specific roles to play during the evaluation. These roles are described in Table 5.

Project decision makers are people who are authorized to speak for the development project or have the right to command modifications to it. This group normally consists of the project manager, the customer who is footing the bill for the development, the architect, and the person commissioning the evaluation.

Architecture stakeholders include developers, testers, integrators, maintainers, performance engineers, users, builders of systems interacting with the one under consideration, and others. Their job during an evaluation is to state the specific quality attribute goals that the architecture should meet in order for the system to be considered a success. This group usually consists of twelve to fifteen people.

Table 5: ATAM Evaluation team roles with their responsibilities [60]

Role	**Responsibilities**
Team Leader	• sets up the evaluation coordinates with client, making sure client's needs are met • establishes evaluation contract • forms evaluation team • sees that final report is produced and delivered (although the writing may be delegated)
Evaluation Leader	• runs evaluation • facilitates elicitation of scenarios • administers scenario selection/prioritization • process • facilitates evaluation of scenarios against architecture • facilitates onsite analysis
Scenario Scribe	• writes scenarios on flipchart or whiteboard during scenario elicitation • captures agreed-on wording of each scenario, halting discussion until exact wording is captured

Proceedings Scribe	• Captures proceedings in electronic form on laptop or workstation, raw scenarios, issue(s) that motivate each scenario (often lost in the wording of the scenario itself), and resolution of each scenario when applied to architecture(s) • also generates a printed list of adopted scenarios for handout to all participants
Timekeeper	• helps evaluation leader stay on schedule • helps control amount of time devoted to each scenario during the evaluation phase
Process Observer	• keeps notes on how evaluation process could be improved or deviated from; usually keeps silent but • may make discreet process-based suggestions to the evaluation leader during the evaluation • after evaluation, reports on how the process went and lessons learned for future improvement • also responsible for reporting experience to architecture evaluation team at large
Process Enforcer	• helps evaluation leader remember and carry out the steps of the evaluation method
Questioner	Raise issues of architectural interest that stakeholders may not have thought of

Appendix 5. ATAM Evaluation Phases and Steps

Partnership and Preparation

First Step
The first step mainly consists of the presentation of the ATAM with its steps and outputs to the three participating groups mentioned above.

Second Step
During the second step the context for the system and the primary business drivers which are the reasons for the system's development are presented to the involved persons. Business drivers are all the functions, information and people enforcing the business goals of an enterprise and ensuring the daily business. Therefore, the system's most important functions, the enterprise's business goals and their relation to the system, any relevant technical, managerial, economic, or political constraints, and the system's major stakeholders are presented.

So actually the desired effect of the system on its environment is described.

Third Step
In the third step, the architecture is presented at an appropriate level of detail that means the presentation is depending on how much of the architecture has been designed and documented; how much time is available; and the nature of the behavioural and quality requirements. The architectural presentation covers technical constraints like the operating system, hardware, or middleware which are intended to be used, and further it shows other systems with which the system must interact. Most important, the architect describes the architectural approaches used to meet the functional and non-functional requirements. The architecture should be described through different views to address different stakeholder roles.

Investigation and Analysis (evaluation)

Fourth Step
During the fourth step the evaluation team identifies the architectural approaches and used patterns and lists them as a basis for further analysis.

Fifth Step
In the fifth step, the quality attribute goals are formulated in detail using a mechanism known as the utility tree. The evaluation team in cooperation with the project decision makers identify, prioritize, and refine the system's most important quality attribute goals, which are expressed as scenarios. The utility tree serves to make the requirements concrete, forcing the architect and customer representatives to define precisely the relevant quality requirements that they were working to provide. A utility tree begins with utility as the root node. Utility is an expression of the overall quality of the system. Quality attributes form the second level because these are the components of utility. Typically, performance, modifiability, security, usability, and availability are the children of utility, but participants are free to name their own as long as they are able to explain what they mean through refinement at the next levels. The third level of the utility tree consists of specific refinements of the quality attributes, for example, performance might be decomposed into *data*

latency and *transaction throughput*. These refinements are the base for the creation of scenarios which form the leaves of the utility tree and they are concrete enough for prioritization and analysis. According to [74], scenarios are the mechanism by which broad and ambiguous statements of desired qualities are made specific and testable. ATAM scenarios consist of three parts:

- *stimulus* which is an event arriving at the system, the event's generator and handler are also named
- *environment* (what is going on at the time)
- *response* (system's reaction to the stimulus expressed in a measurable way)

The definition process of a utility tree is similar to the definition of a quality model for a software product [9] because the overall quality is divided into quality characteristics which are refined in measurable quality attributes which are evaluated by metrics. So metrics are the leaves in a quality model. In the utility tree, scenarios are indicators of certain quality attributes. Of course, a metric is much more concrete because it is a value assigned to an attribute, the scenario in contrast serves to evaluate theoretically whether it is implemented by the architecture. Some scenarios might express more than one quality attribute and so they might appear in more than one place in the tree. To simplify the analysis, these scenarios should be spitted according to different concerns. The refinement process of quality attributes to scenarios might lead to many scenarios which cannot all be analyzed, so this fifth step also includes the prioritization of the scenarios.

This prioritization can be based on a scale from zero to ten or on a relative ranking like high, low, and medium. The latter one is recommended by [74] because it is less time consuming. The ranking is done by the project decision makers. Furthermore, the scenarios are prioritized by the architect regarding the difficulty of satisfying the scenario by the architecture. There also the high, medium, and low ranking is recommended. Now each scenario has an associated ordered pair (importance of scenario for the system, difficulty of satisfying the scenario by the architecture), for example (H,H). The ordered pair (H,H) means, this scenario is very essential for the system and it is difficult to implement it by the software architecture.

The scenarios that are the most important and the most difficult will be the ones where precious analysis time will be spent, and the remainder will be kept as part of the record. A scenario that is considered either unimportant (L,*) or very easy to achieve (*,L) is not likely to receive much attention. The output of utility tree generation is a prioritized list of scenarios that serves as a plan for the remainder of the ATAM evaluation. It tells the ATAM team where to spend its (relatively limited) time and, in particular, where to probe for architectural approaches and risks. The utility tree guides the evaluators toward the architectural approaches for satisfying the high-priority scenarios at its leaves. The utility tree for the ATAM evaluation of video conferencing protocol architecture is shown in Figure 10.

Sixth Step

The following sixth step contains of the analysis of the architectural approaches. The architect explains how the high-ranked scenarios are implemented by the architecture and the evaluation team documents the relevant architectural decisions and identifies and catalogues their risks, non-risks, sensitivity points, and tradeoffs. The architect has to explain which approaches and architectural decisions meet the quality requirements. The upcoming discussion leads to deeper analysis, depending on how the architect responds. The key is to elicit sufficient architectural

information to establish some link between the architectural decisions that have been made and the quality attribute requirements that need to be satisfied. At the end of this step, the evaluation team should have a clear picture of the most important aspects of the entire architecture, the rationale for key design decisions, and a list of risks, non-risks, sensitivity points, and trade-off points.

Testing

Seventh Step

The seventh step is stakeholder-oriented because the evaluation team asks the group of stakeholders to brainstorm scenarios which are operationally meaningful regarding the stakeholders' individual roles. These scenarios are also prioritized because of the limited time for analysis. First, stakeholders are asked to merge scenarios they feel represent the same behaviour or quality concern. Then they vote for those they feel are most important.

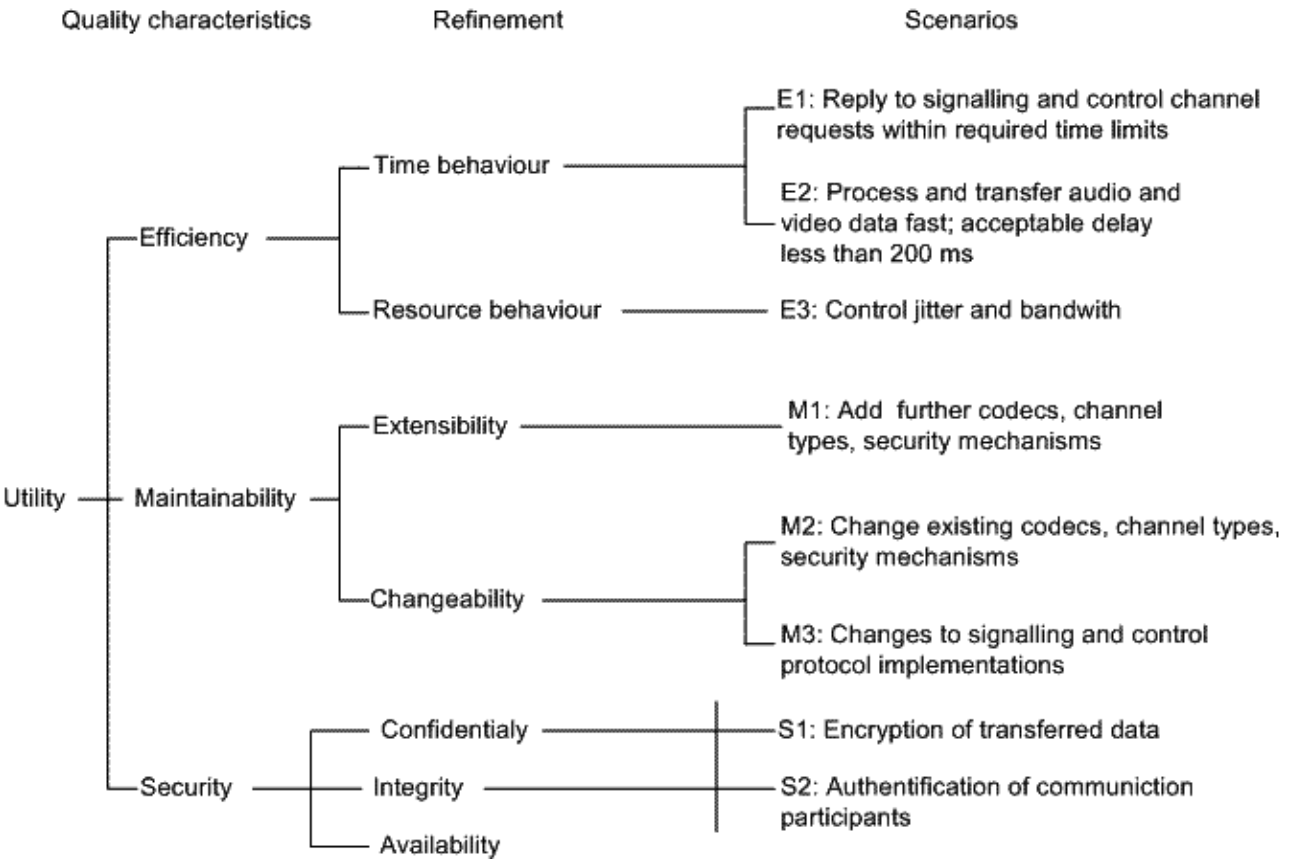

Figure 10: Utility tree for ATAM evaluation

Eighth Step

In the eighth step the architect explains to evaluation team how relevant architectural decisions contribute to realizing each of the chosen scenarios from step seven. During the architect's explanations the evaluation team again identifies and catalogues risk, non-risks, and trade-offs.

Reporting

Ninth Step

Then, in the ninth step, the gained information from the ATAM needs to be summarized and presented once again to stakeholders.

- ATAM's evaluation phase results in the following outputs:
- architectural approaches documented
- set of scenarios and their prioritization from the brainstorming
- utility tree
- risks
- non-risks
- sensitivity points and trade-off points

Finally, the evaluation team groups risks into risk themes. For each risk theme the affected business drivers from the second step are identified. By relating risk themes to business drivers the risk becomes also tangible for non-technical stakeholders like managers.

Appendix 6. Cost Benefits Analysis Method (CBAM) Inputs, Evaluation Steps and Evaluation Roles

Prerequisites and Inputs for CBAM

Since CBAM is building on the ATAM this implies that there will be necessary some prerequisites like:

- Architecture accommodation and presentation necessary for all participants
- Familiarity with concepts like sensitivity points, trade-off points, descriptive scenarios and requirements elicitation where necessary

Inputs in a CBAM evaluation session are:

- Business goals presentation
- Architectural decisions and possible trade-offs (results of ATAM)
- Quality attributes expectation level and economical constraints
- Templates and guidelines for supporting the descriptive scenarios' generation process can be provided.

The architecture ATAM evaluation is also considered input for CBAM.

Steps in a CBAM Evaluation Session

CBAM consists of two phases. First phase is called triage followed by a second phase called detailed examination. The first phase is sometimes necessary in case there are many architectural strategies to be discussed and just a few must be chosen for further detailed examination. Else the evaluation process starts right form the second phase. For both phases in CBAM are prescribed six main steps:

Step 1: Choose Scenarios of Concern and their associated Architectural Strategies
In the first step are chosen the scenarios that concern most the system's stakeholders.
For each of these scenarios there are proposed different architectural strategies that address the specific scenarios.

Step 2: Assess Quality-Attribute Benefits
In the second step are elicited the quality-attributes benefits form participating managers who best understand the business implications of how the system operates and performs.

Step 3: Quantify the Benefits of the different Architectural Strategies
In the third step are elicited the architectural strategies from the participating architects who understand how a certain architectural strategy can achieve the desired level of quality.

Step 4: Quantify the Architectural Strategies' Costs and Schedule Implications
In the fourth step are elicited the cost and schedule information form the stakeholders (both business managers and architects). The evaluation team assumes that within the organization already exists enough experience in estimating time schedules and associated costs.

Step 5: Calculate Desirability

Based on the elicited values resulted in the previous step, the evaluation team the desirability level for each architectural strategy based on the ratio "benefit divided by cost". Further more there is calculated the uncertainty associated with these values, which helps in the final step of making decisions.

Step 6: Make Decisions

Based on the values resulted in step five and the degree of realism of these values there are chosen the best cost-benefit effective architectural strategies which can fulfil best the elicited descriptive scenarios.

CABM Roles

There are three classes of roles participating in CABM:

- *External stakeholders* are having no direct involvement in the software architecture development process. They are the system's stakeholders and their role is to present the project business goals, provide the system quality attributes and their expected level of achievement in a measurable way, and assess the CBAM evaluation results. Examples of external stakeholders are business management team, project management, etc.
- *Internal stakeholders* are having a direct involvement in proposing software architectural strategies that can meet the quality requirements. They have the role of analyzing, defining and presenting the architectural concepts estimating the costs and schedule and uncertainty associated with these strategies. Examples of internal stakeholders are the software architects, system analysts or the architecture team.
- The *CBAM team* has no direct stake in the system's software architectural strategies but conducts the CBAM session. They the role of supporting the system's stakeholders presenting the business goals as such as after the presentation the system's significant quality attributes and their associated scenarios can be easily elicited and formulated. CBAM team also supports the architecting team in addressing the architectural strategies able to satisfy the quality scenarios and estimate the costs, benefits and time scheduling associated with these strategies. CBAM evaluation team consists of an evaluator (team leader or spokesperson), application domain experts, external architecture experts, and a secretary if necessary.

Appendix 7. Examples of Architectural (Design) Metrics

Three common architectural metrics are cohesion, coupling and the Cyclomatic Complexity. They are briefly described in the following.

Cohesion describes the dependencies between methods within a single software component to fulfil a single and precise task. So a high cohesion means that all parts of a component are necessary for fulfilling the task. *Coupling* regards the dependencies between different components. The lower the coupling the more independent are the components from each other and the easier are changes to the system. For many systems, an architecture is desired which aims on a maximal cohesion and a minimal coupling because that supports the system's maintainability. An example of measuring the coupling between modules of software system is given in [75].

Another import metric is the *Cyclomatic Complexity*. According to [76], the Cyclomatic Complexity of a method is the count of the number of paths through the method's source code. Cyclomatic Complexity is normally calculated by creating a graph of the source code with each line of source code being a node on the graph and arrows between the nodes showing the execution pathways. An implementation with a high Cyclomatic Complexity tend to be more error-prone, difficult to test with high coverage, and also more risky regarding maintainability (especially for changeability).