ROYAL INSTITUTE
OF TECHNOLOGY

ENTERPRISE ARCHITECTURE FOR IT MANAGEMENT
A CIO DECISION MAKING PERSPECTIVE ON THE ELECTRIC POWER INDUSTRY

Mathias Ekstedt

September 2004

Submitted in partial fulfillment of the requirements for the
degree of Doctor of Philosophy

Industrial Information and Control Systems
KTH, Royal Institute of Technology
Stockholm, Sweden

Ex.R. 04-02
TRITA-ICS-0402
ISSN 1104-3504
ISRN KTH/ICS/R--04/02--SE

Cover photography © 2004 Mathias Ekstedt

Stockholm 2004, Universitetsservice US AB

Abstract

Within the electric power industry, the average company's enterprise system - i.e. the overall system of IT related entities - is today highly complex. Technically, large organizations posses hundreds or thousands of extensively interconnected and heterogeneous single IT systems performing tasks that varies from enterprise resource planning to real-time control and monitoring of industrial processes. Moreover are these systems storing a wide variety of sometimes redundant data, and typically they are deployed on several different platforms. IT does, however, not execute in splendid isolation. Organizationally, the enterprise system embraces business processes and business units using as well as maintaining and acquiring the IT systems. The interplay between the organization and the IT systems are further determined by for instance business goals, ownership and governance structures, strategies, individual system users, documentation, and cost.

Lately, Enterprise Architecture (EA) has evolved with the mission to take a holistic approach to managing the above depicted enterprise system. The discipline's presumption is that architectural models are the key to succeed in understanding and administrating enterprise systems. Compared to many other engineering disciplines, EA is quite immature in many respects. This thesis identifies and elaborates on some important aspects that to date have been overlooked to a large extent. Firstly, the lack of explicit purpose for architectural models is identified. The thesis argues that the concerns of a company's Chief Information Officer (CIO) should guide the rationale behind the development of EA models. In particular, distribution of IT related information and knowledge throughout the organization is emphasized as an important concern uncared for. Secondly, the lack of architectural theory is recognized. The thesis provides examples of how theory, or analysis procedures, could be incorporated into the Enterprise Architecture approach and hereby concretely drive the development of the architectural models. Due to the nature of enterprise systems, EA theories inevitable will be of an indicative character. Finally, in relation to the models as such, three aspects are highlighted. Firstly, the cost of collecting information from the organization to populate models is routinely neglected by the EA community. This expense should be evaluated in relation to the utility of analyses that the information can provide in terms of better informed decision making by the CIO. Secondly, models (and metamodels) must be kept consistent. And thirdly, the design of models is restricted by the limited mental capabilities of the minds of the model users. CIO concerns must consequently be easy to extract from the Enterprise Architecture models.

Key words: Enterprise Architecture, Enterprise System, Chief Information Officer (CIO), Information Technology (IT) Management, Architectural Theory, Electric Power Industry.

Acknowledgements

When I received my M. Sc. and started thinking about where to go next in life, I read in one of those student magazines that the best trick for becoming successful is to choose the right people to work with. Now, almost five years later, when I am looking at a finished Ph. D. thesis lying in front of me, I completely agree with that magazine. Not only is the graduation as such a big achievement for me, but more importantly has the journey towards the goal been an exciting and rewarding time. The success then, for me personally, thus lies in the means as well as the ends, and without a doubt, neither would have turned out so well to my delight without the individuals that have been around me these years. Frankly, not even the very decision of choosing (the superb alternative of) research studies can I claim to be completely my own!

First of all I would like to thank my supervisor Professor Torsten Cegrell; both for letting me in into the ICS department family in the first place, but also of course for the guidance along the way of growing up academically. No happy family without a mother! In the case of ICS, this role is Mrs Judy Westerlund's. Thank you for always, always spreading a warm and happy atmosphere! Continuing the analogy, next in line is the older brother. Thank you Dr. Pontus Johnson for being a terrific colleague and my guide and academic role model, thank you also for the good laughs, all the discussions about the big and the small and the 'factors', and most importantly, for being a good friend. Among the other sisters and brothers of the department I would specifically like to thank: Dr. Jonas Andersson, Lic. Eng. Erik Johansson, Lic. Eng. Joakim Lilliesköld, and Ms. Åsa Lindström for excellent cooperation and inspiration as well as for amusing and interesting discussions.

There does exist a life outside the university, and furthermore I am of course also grateful towards my biological family; Mum, Dad and Andreas, thank you for your encouragement and for keeping my feet on the ground! Finally, my beloved soon-to-be-wife Nina; thank you for being there and supporting me always, even though I have been absent both physically and mentally many times.

Stockholm, September 2004

Mathias Ekstedt

List of Included Papers

Paper A: Ekstedt, M. and P. Johnson, "Exploring Architectural Analysis Credibility from a Developer Perspective," *In the Proceedings of the Fourth Australasian Workshop on Software and Systems Architecture*, Sydney, Australia, 2002.

Paper B: Ekstedt, M., P. Johnson, Å. Lindström, M. Gammelgård, E. Johansson, L. Plazaola, and E. Silva, "Consistent Enterprise Software System Architecture for the CIO: A Utility-Cost Based Approach," *In the proceedings of the 37th Hawaii International Conference on System Sciences*, Big Island, Hawaii, U.S.A., 2004.

Paper C: Johnson, P., M. Ekstedt, E. Silva, and L. Plazaola, "Using Enterprise Architecture for CIO Decision-Making: on the Importance of Theory," *In Proceedings of the Second Annual Conference on Systems Engineering Research*, Los Angeles, U.S.A., 2004.

Paper D: Ekstedt M., P. Johnson, and P. Sjölin, "The Architectural Information View for the Power Electricity Industry," *In Proceedings of CIGRÉ SC D2 Colloquium*, Rio de Janeiro, Brazil, 2003.

Paper E: Ekstedt, M., P. Johnson, J. Lilliesköld and N. Jonsson, "Making Project Complexity Understandable: the Elegance of Notations," *In Proceedings of the 12th International Conference on Management of Technology*, Nancy, France, 2003.

Table of Contents

[W]e require that the land as well as the inhabitants […] should be taken in at a single view,
for a country which is easily seen can be easily protected
Aristotle

Simplification may lead to error,
but there is no realistic alternative in the face of the limits on human knowledge and reasoning
Herbert A. Simon

Chapter 1

Introduction to Enterprise Architecture

This thesis is the resulting work from almost five years of fostering at the department of Industrial Information and Control Systems at Royal Institute of Technology (KTH) in Stockholm, Sweden. It adheres to the fairly new and quickly expanding discipline of Enterprise Architecture. Even though here presented under a new name, it is still much the same topics that been around at the department since long. And concretely, this work has been inspired by, and is in many senses a continuation of, the previous dissertations of Johnson (2002), Andersson (2002), and Haglind (2002).

This introductory chapter covers the basic elements of Enterprise Architecture, and thus the specific theoretical and empirical background on which the present research is based. For the reader familiar with Enterprise Architecture this chapter will cover no news, merely give those readers an understanding of how the author has chosen to prioritize within the field, for better or for worse.

The chapter starts with a description of the background for the work in terms of the addressed part of the real world – labeled the enterprise system, as well as the prime actor when it comes to managing this system – the Chief Information Officer (CIO). Subsequently, the academic belonging of this work is covered. Primarily the discipline of Enterprise Architecture is outlined, but other related works that has inspired and influenced the research are also briefly introduced. The chapter closes with a summary of the main contributions of the thesis.

1.1 BACKGROUND

This section describes the empirical piece of the world that is tackled within the present research. Firstly, the concept *enterprise system* is introduced for reflecting all the phenomena in a company that has to do with information technology (IT). Secondly, the characteristics for the overall management of this system are described by portraying the role of the Chief Information Officer. These two sub sections jointly define the scope of the work.

When it comes to experience, the forthcoming description is one of the electric power industry, since it is within this particular domain that the present research has been empirically conducted. However, it is the belief of the author, as well as the majority of literature, that the picture is remarkably similar in most industrial domains.

1.1.1 Enterprise Systems[1]

Contemplating a company as a whole, there are immensely many things and phenomena that attract attention. Disregarding all the aspects that do not primarily have to do with IT, the scene becomes clearer, but there are still too many phenomena to have a comprehensible picture. The image will contain a vast amount of different things, for instance; IT (or software) systems, which perhaps may be seen as the heart of the computerized company; data (also referred to as information), which ranges from records of customers to technical details of used machinery; functionality, which is the active manipulation of data and performance of many work tasks; interfaces and connectors, which connects software systems so that data and functionality can be used in a more boundless manner; users, the individuals interacting with the IT systems and (hopefully) benefits from them; organizational units, which are using, owning and maintaining the IT systems; business processes, which are a value oriented set of activities in the company; goals and visions, which guide the overall company engagement; strategies, governing how the goals are to be achieved for various levels and domains; projects, which organizes how specific efforts around IT is to be performed; knowledge, the skills and know-how within the company spread among the personnel; and much more. This list is not complete; rather it aims at demonstrating the scope of IT related issues within a company. All of the above mentioned entities are, as indicated, not acting in splendid isolation, they are interconnected in various and plentiful, more or less obvious, manners. And increasingly so over the years it seems.

Technical Aspects

Technically, some thirty or forty years ago, IT was typically oriented in so called stove-pipes. The IT systems were implemented for single purposes and were supporting only one organizational unit. Especially, the gap between different stove-pipes has been wide between the industrially oriented systems and administrative systems. E.g., for a distribution

[1] In the included papers, and otherwise in literature, the here called "enterprise system" have been labeled with various names; the enterprise software system, the company-wide software system, the enterprise information system, the enterprise IT system, or simply the enterprise to mention a few. (Occasionally, 'architecture' is confusingly used for this *reality*, but mainly that word refers to *models* of the reality.) This sprawling taxonomy does (primarily) not reflect any real disagreement regarding semantics; rather it is a sign of a young phenomenon with the lack of an established unified academic discipline. In general, however, the here presented denotation of the concept enterprise system adheres to the more broad and including variants of its usage.

network operator in the electric power industry the core of the industrial systems are the real-time SCADA (Supervisory Control And Data Acquisition) systems for local and central monitoring and control of the power grid. Other industrial systems with less extreme performance requirements include systems for geographical information, planning and maintenance, and distribution and production management. On the other side of the gulf the administrative systems covers support for customer relationship management, billing, pay roll management, financial accounting, sales and marketing, and much more. In order to better achieve business goals, systems have during the years been added, extended, and, most importantly, integrated into a system in its own right; the enterprise system. In large organizations several thousands of interconnected single IT systems may be employed. The size of each single system may vary extensively from very large enterprise resource planning systems to small custom-made niche products. Unfortunately, these emerging enterprise systems have typically not evolved through a careful and a holistically planned approach, the stove-pipe mentality often survived longer than the stove-pipe architecture. Due to all its legacy systems, the enterprise system is composed of a considerable number of heterogeneous and poorly understood components executing on a diverse set of platforms. The interactions between the individual systems have in many cases become quite extensive and typically this communication is performed on an equally wide range of connectors utilizing many different technologies. The complexity is increased even more by the fact that many of the different systems are storing redundant data and implement rather like functionality. Not to mention the sheer amount of data records and functionality that is included in the overall enterprise system.

An evident treatment to such diverse enterprise systems is of course standardization. The trend has, for instance, also been to integrate small subsystems into larger homogenous groups of prepackaged systems from a single vendor. Especially within the domain of enterprise resource planning systems has this advancement been strong. The tendency is however far from clear-cut. For several reasons the opposite decentralized approach to enterprise system design is also widespread. Then the integration mechanisms become central since the individual IT systems are rather considered as the variables and the integration solutions as the surviving constants. This evolution is for instance reflected in the topics of enterprise application integration and middleware, which have gained much attention over the last decade.

THE USER ORGANIZATION

Organizationally, the companies have become more integrated over the years as well. And actually, the causality is typically as much oriented in the opposite direction too; increasing organizational integration drives technical integration, as well as vice versa. Perhaps the single most influential contributor to organizational integration is the introduction of the concept of business processes and the efforts of business process reengineering (Davenport 1993, Hammer and Champy 1994). Instead of having a function oriented company, the explicitly value-producing and customer-oriented business processes became the focal point for business design. And a means for successful business processes design was, and remains, intelligent use of IT systems. The functional units however still remain and they typically hold responsibilities of the different sub activities of the process. For instance, a process such as maintenance within a power company can be found ranging over organizational units such as customer call centers, control centers, field service units, and economy units. Processes are typically hierarchically organized, and in large companies on mid-level granularity up to hundreds of processes are found. The organizational aspects of the enterprise system, however, reach further than only business processes and functional units. The design of processes and units are for instance motivated by business goals. On a high level such goals define ambitions for revenue, and further down they outline what market segments, what products, etc. that the company should engage in. Another concern is how these goals are to be accomplished, i.e. the area of strategy and planning. Particular interest is often directed towards strategic alignment between business and IT operations. Finally, of course there are numerous users interacting with the IT systems that are completely dependent on the systems in order to perform their allotted responsibilities. These aspects and assets (and more) are all organizational facets included in the enterprise system.

The Maintenance Organization

From an overall perspective on the organization, governance structures in relation to its IT systems is key to the evolution of the enterprise system; who owns what IT systems and who are authorized with what decisions? From a more technical perspective, the IT department (or the like) and the processes related to IT management and maintenance are of course especially important. Since the nature of the enterprise system has changed and extended over the years, so have the competencies of the IT personnel. Primarily, the trend has gone from in-house and custom-made development of IT systems to acquisition of standard (sometimes off-the-shelf) systems. Instead of mainly building new systems automating manual work, much effort is today spent on improving the current IT system portfolio. Perhaps oversimplifying, one might say that the former tasks of designing and analyzing single systems, eventually employed on one execution platform, have now been replaced by acquiring, integrating, and phasing out IT systems in the heterogeneous enterprise system. In the wake of this trend, the issue of sourcing has also become a major issue. Not only is this a matter of governance structures for the systems as such, but also is it influencing the (IT) personnel; what competences should be kept within the company and what should be externally bought?

Thus, the tasks modern IT department ranges from hands-on software coding and configuration to project management, requirements engineering and contract management.

1.1.2 The Chief Information Officer

The purpose of the previous section was to depict the real-world situation (as opposed to any models of it) of enterprise-wide IT concerns. From the description it should be evident that taking on the responsibility of the overall IT management at a company is an immense challenge; this is the work of the *Chief Information Officer* (*CIO*).

CIO CONCERNS

The CIO role emerged in the 1970s and is typically held by a small group of people or an individual, close to the senior management of the company (Gottshalk and Taylor 2000). The authority of the CIO differs with company, but generally it is the role responsible for all the IT in the organization taken as a whole; i.e. the topic of the previous chapter. The primary focus of the CIO is of strategic character for planning of IT systems of the enterprise system. As indicated in the previous section, the responsibilities thus cover a broad technical and organizational scope. The particular concerns of a CIO's agenda typically include; IT and business alignment, i.e. how well the IT systems support the business of the company; IT investment decisions, e.g. what systems that should be prioritized for acquisition; IT system quality assessment and improvement, covering aspects such as security, performance, reliability, integrability, and maintainability; organization of IT personnel, i.e. managing the company knowledge so that synergies can be achieved; IT governance, i.e. stipulating what types of decisions that are to be taken by whom; development of IT strategies, guiding further IT-system development; and managing IT costs, both in terms of the systems operations as such and the personnel expenditures.

CIO concerns constitute the driving force behind the present research and the CIO consequently represents the ultimate "customer" of this work. Obviously, the interest and the situation of the CIO is a truly empirical question that cannot be deduced from theory. It have been reported on in literature (Boar 1993, Frenzel 1996, Ward and Griffiths 1996, Gottshalk and Taylor 2000, Kirkland 2002, Kirkpatrick 2002, CIO Council 2004), but still remains to be much more explored. Especially this need is apparent when playing the role of being the primary driver for Enterprise Architecture research.

THE CIO AS A DECISION MAKER

From a more theoretical perspective (as opposed to the above where the empirical substance was the topic), the CIO can be described as a decision maker with the mission to make rational choices in an overwhelmingly information laden world. Take for instance the frequently occurring situation when a new system is to be acquired. Even if the choice can be reduced to identifying the "best" out of only two systems, this is generally an extremely complicated undertaking (for a rational actor). First of all it is rarely apparent what the criteria for being "best" are, but even when this is agreed on, identifying and collecting information needed for doing this assessment is often very resource demanding (and as a rule practically impossible.) Consider for instance the challenge in assessing only the three aspects of security level, maintainability, and alignment between IT and business as indicators of being "best."

Two delimitations have permeated the present work when it comes to CIO decision making; the bounded rationality of a single mind and the consequence of paucity of relevant information. These concepts, taken from decision making theory (Simon 1997, March 1994, Rubinstein 1998, Hastie and Dawes 2001), are elaborated on below.

Complexity and Rationality

Complexity paralyses the human mind. This is uncontroversial. But what complexity is, however, is a matter of dispute. Given this, the subject of complexity is an academic mine field, and it is not the ambition of the current research to contribute to the area as such, rather the aim is to find a relevant and pragmatic description for the environment of the CIO. Without further elaboration on the exact definitions of complexity, this thesis simply stipulates the enterprise system is complex and leave to the reader to give the word full connotation. (As the saying puts it; you recognize the elephant when you see it, even though it might be hard to describe.)

This thesis has approached the tricky complexity concept by turning to the term *bounded rationality* coined by Herbert Simon (1997a and 1997b) for describing human behavior. The description of humans being boundedly rational came as a reaction to the model of humans as rational actors used in economic theory, and it highlights the practical limitations of rational choice. The rational actor (in our society considered the role model) is supposed to act according to a logic guiding of three questions; “what is feasible?”, “what is desirable?”, and “what is the best alternative according to the notion of desirability, given the feasibility constraints?” (Rubinstein 1998). Rational and actual behavior differs for instance due to the fact that the decision maker does not have a complete knowledge of the consequences of each choice, and rationality requires a choice among all the feasible alternatives but only few of these ever comes to mind (Simon 1997). Simon called the actor of bounded rationality the “administrative man”. The CIO will always be an administrative man, not by failure, but by laws of nature. Accepting this fact, the challenge for research is to serve the CIO with limited information with as high relevance as possible, not *everything* of *some* importance. The assumption in the present research is that, due to the bounded rationality, complexity that could be managed by an individual is in principle constant. However, complexity of a given problem is influenced not only by the mental capabilities of the human facing the problem, but also the representation and existing knowledge of the problem. If we can describe, in theory and model, the problem in a coherent way (so that it is easy to understand), the problem becomes less complex (cf. King et al. 1994). Consider for instance the difference in complexity of a problem a physicist faces related to gases, depending on if every molecule has to be described individually or if the (simplified) statistical thermodynamics of Maxwell and Boltzmann is available. Paper E elaborates on how complexity varies with notation syntax of models (since architectural models are the base of the EA discipline.)

It is not only the limitations of the single mind that promotes a careful information selection process. Also, the fact that each piece of information costs to acquire is a driving force. Or in the words of March (1994): “Decision makers try to understand their environments on the basis of inadequate information. History is not generous with observations. Sample size is small.” In the case of the CIO, an unkind history is perhaps not the prime reason for poor decision support since the company is filled with all relevant information, which in principle is possible to elicit and deliver to the CIO. However, the time and money that must be spent doing so can easily exceed the value of taking that piece of information into consideration in the current decision. The topic of information collection cost is further elaborated in paper B, and in paper D circumstances for information searching is illustrated in a more with a case study from the electric power industry.

1.2 RELATED WORKS

Given the chaotic real-world of enterprise systems and the interests and limitations of the CIO, there should be little doubt that the CIO is in need of management and comprehension support for his undertaking. Enterprise Architecture has been proposed as a discipline to the rescue, and that is the topic of the first part of this section. However, Enterprise Architecture is not a discipline mature enough to be the only academic background to this work, so a brief sub section outlining other influential sources follows the first one.

1.2.1 ENTERPRISE ARCHITECTURE

To a large extent Enterprise Architecture (EA) has been deduced as a response to an ever increasing need of abstracting software systems. So, the fifteen year old claim of Mary Shaw (1989) is still true today; “Larger Scale Systems Require Higher-Level Abstraction.” This was a call for software architecture, and now it has to be repeated for the enterprise level of system architecture. (The statement was of course true long before Shaw phrased it, and it is interesting to contemplate the similarities between the arguments of today with for instance the, another fifteen years older, article of Niklaus Wirth (1974) on structured programming.) Enterprise Architecture has evolved for other reasons too. Over the years it has become evident that the quality and appropriateness of a system cannot be assessed only with the system itself, its context must be taken into consideration. In the case of the continuously increasing number of software systems of user-side companies (rather than vendors), this context is the one of the business organization and the IT departments (as depicted previously). Those areas have been included in EA, to a lesser or greater extent, alongside of the pure technical issues. Consequently, architecting in the enterprise setting is both a technical and an organizational undertaking.

Considered as a discipline in its own right, Enterprise Architecture is extremely young. The first widely referenced paper is Zachman’s IBM Journal article from 1987 (Zachman 1987), and the first book extensively referenced is the work of Spewak from 1992 (Spewak 1992). However, the EA problems and concerns indeed have been around longer than the late 1980s, then under disciplines such as Strategic Information Systems Planning. A good fifteen years later, the academically newborn discipline of course still struggles with defining its foundations. Pragmatically, Enterprise Architecture is defined[2] by its supporters, who are found in literature (Sowa and Zachman 1992, The Clinger-Cohen Act 1996, Boar 1999, Armour et al. 1999a, Armour et al. 1999b, CIO Council 1999, Boster et al. 2000, Armour and Kaisler 2001, Armour et al. 2002, The Open Group 2002, United States' Department of Defense 2003, Jonkers et al. 2003, Perks and Beveridge 2003, Wegmann and Preiss 2003, O'Rourke et al. 2003, McGovern et al. 2004), and on the web (CIMOSA 2004, EAC 2004, FEAPMO 2004, GEAO 2004, IFEAD 2004, Ronin 2004, ZIFA 2004).

As mentioned previously, it is the presumption of the present research that the CIO (as depicted above) is the stakeholder amongst others whose needs should drive and guide the Enterprise Architecture discipline. This is not always an explicit statement in EA references, and could perhaps therefore be questioned by some.

Frameworks

What perhaps come closest to defining the area of EA today are its various frameworks. Among these different frameworks consensus seem to be prevalent on taking a model-based approach to the mission of enterprise system management. Architectural models such as drawings, maps, etc., constitute the assumed key to success. Basically, EA models describe high level abstractions of entities of the enterprise system and how they relate to each other. I.e., this scope includes technical entities such as data, functionality, physical infrastructure, applications, and interfaces, as well as organizational entities such as business processes, goals, organizational units, governance structures, and work flows. Furthermore, issues such as technology reference models, standards profiles, and methodology for architecting, are all part of EA frameworks. Support can also be found on how the entities of the EA models can be refined into more detailed designs. There is however no single model type for EA and many notations, such as the Unified Modeling Language (UML) (OMG 2003), as well as many idiosyncratic notations, are used for modeling the above described semantics.

Developing frameworks seem natural in a model-based discipline, and this is also how it once started. In the previously entitled first paper of EA, Zachman introduced, as the title clearly mediates, "A framework for information systems architecture." Since then several other frameworks have materialized. One might think that all of these frameworks are competing for survival on the profit of each other. This seem however not to be the case, at least at the moment. Today they are to a large extent fairly good complements to each

[2] Just as there is a problem with the consensus in the choice of word for the elaborated piece of reality, here labelled the enterprise system, likewise are there many words for the models and the discipline describing it. Enterprise Architecture is pretty much standard these days, but words such as Enterprise IT Architecture, Enterprise Information System Architecture, Information System Architecture, Enterprise Software System Architecture, and Enterprise System Architecture is also frequently encountered. For instance, in some of the included papers, Enterprise Software System Architecture is used with the same connotation as the one of Enterprise Architecture in this introductory part of the thesis.

others. Below, the frameworks that largely have influenced the present research are introduced with their primary respective foci.

The U.S. Department of Defense Architecture Framework (DoDAF) is scoped for war fighting operations and business operations and processes and is a critical mechanism in the Net-Centric Operations and Warfare (NCOW) initiative (DoD 2003). The framework does in many ways nevertheless serve well as a general purpose construction for EA. The core of DoDAF is its products. These are architectural models in which entities are depicted graphically, textually, or in tables. On a high level these products are divided into three categories: the Operational View (OV) products, focusing on operational needs and who does what; the System View (SV) products, relating systems and characteristics to operational needs; and the Technical Standards View (TV) products, which prescribe standards and conventions. All in all, the DoDAF introduces twenty six different models that together serve to support the enterprise systems management. The current DoDAF is an evolution of the Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) Architecture Framework of 1997 (DoD 1997).

The Open Group is a vendor-neutral international consortium with more than two hundred member companies, that among other things are promoting a framework for EA; The Open Group Architecture Framework (TOGAF) (TOG 2002). In contrast to DoDAF, the latter framework focuses on the process of building and evolving Enterprise Architectures, rather than the syntax and semantics of the EA models. The fundament of TOGAF is the Architecture Development Method (ADM), which describes "a method for developing an enterprise architecture to meet the business and technology needs of an organization." The ADM is a generic and iterative development cycle with eight main phases (and one preliminary phase); Architecture Vision, Business Architecture, Information System Architectures, Technology Architecture, Opportunities and Solutions, Migration Planning, Implementation Governance, Architecture Change Management. Each of these phases is further detailed into sub steps and advice on modeling tools and techniques are also given here. Apart from ADM, two other parts are prominent; the Enterprise Continuum, illustrating how general purpose architectures are developed to organization-specific architectures; and the Resource Base, providing a set of tools and techniques for applying the TOGAF concept.

The United States' Office of Management and Budget (OMB) have established the Federal Enterprise Architecture Program Management Office that is developing the Federal Enterprise Architecture (FEA) (FEAPMO 2004). This framework is a collection of five reference models with different perspectives of an enterprise system: the performance reference model (FEAPMO 2003a), describing measures for the performance of IT initiatives; the business reference model (FEAPMO 2003b), describing business operations of the federal government; the service component reference model (FEAPMO 2003c), classifying the IT services; the technical reference model (FEAPMO 2003d), describing standards, specifications, and technology; and finally, the data and information reference model (yet to be released) intended to describe the information and data supporting the operations. A strong point in FEA is the extensive standardized taxonomy the reference models provide.

Perhaps the most extensive, and probably the most well-known, framework is Zachman's (Zachman 1987, ZIFA 2004). Through the years it has been extended and today it is summarized as a matrix with six rows and six columns (Sowa 1992). The first five rows differentiates *perspectives* of building (or modifying) an enterprise by roles; *Planner*, *Owner*, *Designer*, *Builder*, and *Subcontractor*. The sixth row covers the final product of the enterprise system and is labelled the *functioning enterprise*. Horizontally, the framework is divided into the six aspects; *what*, *how*, *where*, *who*, *when*, and *why*. Altogether, the Zachman framework thus provides thirty six cells from which an enterprise could be understood and described. A third dimension of the framework, called *science*, has also recently been proposed (O'Rourke et al. 2003). This extension is known as the Zachman DNA (Depth iNtegrating Architecture). In addition to the perspectives and aspects the z-axis is used for classifying the practices and activities used for producing all the cell representations. The Zachman framework is neutral to tools and techniques. It does not provide support for how to go about Enterprise Architecting; rather it is classification schema for organizing descriptive representations of an enterprise system. Perhaps its strongest benefit lies in serving as a vehicle for communication.

Many other, more or less extensive, frameworks do also exist, and many are variants of the ones presented above. For instance; the U.S. CIO Council's Federal Enterprise Architecture Framework (FEAF) (CIO Council 1999), Spewak's framework for Enterprise Architecture Planning (EAP) (Spewak 1992), Armour et al.'s work implemented at U.S. Department of the Treasury (Armour et al. 1999a, Armour et al. 1999b, Armour and Kaisler 2001.), Ronin International's Enterprise Unified Process that is extending the IBM/Rational Unified Process which originally is mainly focusing on single software system development processes (McGovern et al. 2004, Ronin 2004), and the Extended Enterprise Architecture (E2A) Framework from the Institute for Enterprise Architecture Developments (IFEAD 2004).

Viewpoints

According to King et al. (1994), a model in general is *"a simplification of, and approximation to, some aspect of the world. Models are never literally "true" or "false," although good models abstract only the "right" features of the reality they represent.*" And "right" in the context of EA is reflected in the use of the term "viewpoint"[3], illustrating that different models are depicting the *same* piece of reality but from different perspectives for different purposes. Viewpoints of enterprise

[3] Some confusion is usually acquainted to the use of terminology also on this issue. According to IEEE Standard 1471 (2000), *viewpoints* is the definition of the language for describing *views*. I.e., views are concerned with models and viewpoints with meta-models. This use is also the terminology preferred by the author. However, most often this distinction is not made and the two words are used for both of the connotations. This is for instance (unfortunately!) the case in the included papers of this thesis. In addition, other variants of the viewpoint concept are also flourishing, e.g., Clements and al. (2003) use the term viewtype for denoting a set of views.

and software architecture are often illustrated by an analogy back to the origin of building architecture. The electrician and the plumber need different architectures (models) of the same piece of reality (i.e. the house) in order to perform their respective work. Likewise is it within Enterprise Architecture; if the task includes IT- system performance issues, then the information that needs to be modeled (i.e. abstracted from the enterprise system) is different than if the task is concerned with IT- system usability or business and IT strategy alignment. Thus, single viewpoints do not fully represent a system's architecture.

Viewpoints are permeating the systems-oriented disciplines, and not surprisingly are they also at the heart of Enterprise Architecture. E.g., at the core of Zachman's groundwork are the cells of a matrix (cf. the previous sub section) and where the columns are easily mapped to different viewpoints. DoDAF and TOGAF are also promoting viewpoints heavily.

Also in Software Architecture are viewpoints[4] fundamental. The beginning of an explicit and extensive usage of viewpoints came with Kruchten's famous article (1995) introducing "4+1" views for software architecture. The four main views presented were the *logic*, the *process*, the *development*, and the *physical* view, and in addition the use of scenarios was promoted as a redundant view tying the others together. Other viewpoints promoted include the ones of Hofmeister et al. (2000); the *conceptual*, the *module*, the *execution*, and the *code* view. Of course, long before mid 1990s the topic was debated in software engineering by for instance Dijkstra (1976), arguing for separation of concerns. Lately, software architecture viewpoints have been summarized and categorized by Clements et al. (2003).

Also in business and organizational modeling viewpoints are found. E.g. Morabito et al. (1999) differentiate among views by the set of fundamental entities that are included in various descriptions of an organization. A very system oriented approach to business modeling is presented by Eriksson and Penker (2000) who introduces the *vision*, the *process*, the *structure*, and the *behavior* view of business.

When it comes to Enterprise Architecture, viewpoints are becoming more sprawling due to its wide scope. A somewhat technically oriented set, but still fairly representative for the discipline, is proposed by Perks and Beveridge (2003), namely; *business process domain*, *functional*, *security*, *management*, *software engineering*, *data management*, *user*, *system engineering*, and *communications* views. A problem that increases when more viewpoints are introduced ant the total scope are added to is the potential inconsistencies among all the models. This topic is further elaborated in paper B.

In summary, the usage of viewpoints is simply introducing separation of concerns to the problem at hand. And this is only necessary due to the fact that the human mind is easily confused since it cannot grasp too much information at the same time.

[4] However, in Software Architecture, *view* is the term dominating the vocabulary.

THE ROLE OF ENTERPRISE ARCHITECTURE

Summing up, it is the presumption of this research that Enterprise Architecture should serve as decision support, primarily for the Chief Information Officer. This decision support does however not come for free. There is quite an extensive undertaking in performing Enterprise Architecting. Moreover, it is not a quick-fix-project granting success for an infinite future; rather it is a continuous activity of long-term strategic character (which however can achieve fast results) (Spewak 1992, Armour et al. 1999b, DoD 2003).

In general, what an enterprise architecture model can contribute with is to bring some order into a world that otherwise seems chaotic. Moreover, maintaining a good enterprise architecture model makes it less complicated to respond quickly to new demands and evaluate potential future solutions. This is due to a better understanding of the current situation. Or in the words of Armour et al. (1999b): "…well, how do you improve something you can't see?" The present state of affairs (and future modifications of it) can also be communicated in a consistent manner to different stakeholders, both within and outside the company. Moreover, information can be reused effectively. When a new improvement project is started, analyses can be done using already collected information and knowledge.

Enterprise Architecture is thus a discipline of as much processes as it is one of a set of models providing the CIO with a cockpit for IT-management.

1.2.2 ADJACENT DISCIPLINES

Even though presented above as a discipline in its own right, Enterprise Architecture can be seen as a mishmash of several other disciplines in cooperation. The trick for EA then boils down to how all of these other theories are to be combined. Below a short account is given of related work that have had impact on the present research, but which is not labeled as Enterprise Architecture. Some of it is already mentioned above. The aim is not to explain the references, merely to give the reader pointers to where the mindset that have created this work can be re-created.

The main influences for this work are found within Software Engineering. General coverage of the subject with somewhat different approaches is for instance found within Pressman (2000), and IEEE's Software Engineering Body of Knowledge (2001). Examining the area closer, influential sub disciplines include for instance: requirements engineering (Davies 1993, McCauly 1996, Kotonya and Sommerville 1998) dealing with the connection to users and their organizations; software design and modeling (Parnas 1972, Wirth 1974, Dijkstra 1976, Brooks 1995), and in particular object orientation (Jacobson et al. 1992, Booch1991, OMG 2003), dealing with the core of software development; development processes and software project management (Brooks 1995, Royce 1970, Boehm 1988, Kruchten 2000, Beck 2000) and the closely related topic of configuration management (Leon 2000, Berczuk and Appleton 2003), both engaging in organizational issues and workflow. In general however, the one (sub) discipline that has had the largest impact on the present work is Software Architecture (and also Component-Based Software Engineering which to a large extent tend to be the other side of the same coin). From Software Architecture concepts such as *component*, *connector*, *view*, and *style* (or *pattern*) is primarily accumulated. Influential references include the work of Shaw (1989), Perry and Wolf (1992), Denning and Dargan (1994), Kruchten (1995), Gamma et al. (1995), Shaw and Garlan (1996), Buschmann et al. (1996), Clements and Northrop (1996), Bass et al. (1998), Baragry and Reed (1998), Szyperski (1998), Garlan (2000), Hofmeister et al. (2000), Metha et al. (2000), Schmidt et al. (2000), Heineman and Councill (2001), Crnkovic and Larsson (2002), and Clements et al. (2003). One of the most stressed benefits of employing Software Architecture design and analysis is the possibility of early assessment of software qualities. Some influential references about software quality attributes and their operationalization are found in Boehm et al. (1978), Oskarsson (1982), Barbacci et al. (1995), Barbacci et al. (1997), Fenton and Pfleeger (1997), Zuse (1997), Spitznagel and Garlan (1998), Briand et al. (1999), Fenton and Niel (1999), Bachmann et al. (2000), Kazman et al. (2001), and Clements et al. (2002). Running somewhat in parallel, is the formalist approach to software engineering. It uses well defined logics as the foundation and covers the whole spectrum from low-level issues to architecture, cf. Wing (1990), Moriconi and Qian (1994), Moriconi et al. (1995), Moormann Zaremski and Wing (1997), Alagar and Periyasamy (1998), Medvidovic and Taylor (2000). This background was early abandoned in the present research in favour to an indicative approach, cf. Paper A.

In addition to these references, covering mainly single system architectures, the growing area of "enterprise-oriented" software engineering, covering topics such as commercial-off-the-shelf (COTS) software, middleware, and enterprise application integration, has of course also been highly significant for the present research, cf. Taylor (1992), Kobryn (1998), Brown et al. (1998), Brown and Barn (1999), Linthicum (2000), Lutz (2000), Brown (2000), Ruh et al. (2001), Britton (2001), Emmerich et al. (2001), Wallnau et al. (2002), Tyndale-Biscoe (2002), Garland and Anthony (2003), and Sessions (2003). These are highly relevant references for technical oriented enterprise architecture efforts. The CIO is in this context considered the system architect, evolving an enterprise system rather than developing a software system. This change in challenge is nicely illustrated by Wallnau et al. (2002): "[c]omponent-based design is a process of exploration, not refinement." This state of affairs derives from the fact that components and connectors to a large extent come prepackaged and are black-boxed with an unknown interior that can only be externally *experienced*. Further elaboration on the main differences between architecture of systems on the enterprise level compared to single systems are found in Johnson (2002) and Andersson (2002).

Enterprise Architecture could not claim to be a discipline in its own right if only Software Engineering were the theoretical base. Much inspiration has also been accumulated from other disciplines, and the closest one is probably Information Systems. The two are at least partially addressing the same questions, but if Software Engineering has started off from understanding program correctness and execution efficiency (Dijkstra 1976), Information Systems research tend to start off from an outside and management perspective thus including how the systems fit into a context. Information Systems is in itself also a cross disciplinary domain ranging from organizational issues to design of information systems and the difficulties of aligning information systems with the business in the short and long run. From the Information Systems field the following work have been inspiration: Boar (1993), Bruzelius and Skärvad (1995), Luftman (1996), Ward and Griffiths (1996), Frenzel (1996), Axelsson and Goldkuhl (1998), Nilsson et al (1999), Morabito et al. (1999), Eriksson and Penker (2000), and Gottschalk and Taylor (2000).

As indicated previously, Decision Making has been a major influence for understanding the situation of the Chief information Officer. References include March (1994), Simon (1997a), Simon (1997b), Rubenstein (1998), Skinner (1999), and Hastie and Dawes (2001).

The final area, from which the present research has been marked, is the very tricky discipline to define of Systems Engineering. Some would argue that EA is a sub discipline of Systems Engineering since it tries to approach non-homogenous systems holistically by combining different engineering disciplines. (Systems engineering incorporates software, mechanical, electrical, and civil engineering, and so on.) Related work within Systems Engineering includes Checkland (1981), Martin (1997), Stevens et al. (1998), Maier and Rechtin (2000), and INCOSE (2000).

1.3 CONTRIBUTIONS

The contributions of the present work are found in the five included papers. Looking at it in retrospect, the conducted research appears quite theoretical in nature. Four papers primarily address the Enterprise Architecture research community and emphasize prerequisites for further research in the area. One paper is practically oriented proposing a new model (for a theoretically neglected area). As expected within engineering disciplines, all of the research has been guided by a highly normative motto, and on the whole it can be summarized in the following six principles (without order of importance):

Concerns of the Chief Information Officer (CIO) should drive the choice of architectural theory which in turn should drive development of Enterprise Architecture models. Today many of the models tend to be developed without a clear purpose. Models are developed and only when ready are they closer examined with respect to what questions they de-facto can answer according to some tacit rule of thumb. This logic should be reversed and analysis procedures made explicit. This thesis further argues that prime stakeholder of Enterprise Architecture is the CIO. (Paper B and Paper C.)

Architectural theory must be of an indicative nature and is thus afflicted with uncertainty. Theory for the enterprise level of system architecture cannot be deduced from an underlying formal theory. This would require a practically unmanageable amount of information, which oftentimes is not even available. As a consequence, Enterprise Architecture theory will always suffer from a certain amount of incredibility. (Paper A)

Enterprise Architecture meta-models must be kept consistent. Today, many EA models are not developed starting out from an overall meta-model. As a consequence different EA models might very well be diverging and incompatible. (Paper B.)

The cost of searching for information must be included in the overall evaluation of Enterprise Architecture initiatives. Today it is implicitly assumed that all the information needed for populating an EA model (and perform analyses) is at hand, or at least is easy to get hold of. The aspect of information gathering cost is unfortunately widely neglected. This cost must be traded-off against the utility that the information can offer in terms of better-informed decisions. (Paper B and Paper D.)

Organizational knowledge and information about the enterprise system should be incorporated into Enterprise Architecture. Keeping track of how information is scattered throughout the company and how credible that information is estimated to be, is of importance in the process of CIO decision making, in particular for information search cost estimations (cf. the previous principle). (Paper D.)

The difficulties of understanding and using models are determined by the expressiveness of the used notations (and the limits of the user's mind). The same amount of information can be expressed more or less easy to understand in relation to what knowledge it is supposed to mediate. This aspect is of particular interest in the information-laden world of EA. (Paper E.)

The important thing in science is not so much to obtain new facts as to discover new ways of thinking about them
Sir William Bragg

Chapter 2

Research Design

This chapter covers the methodological aspects that have guided the present work. The first subsection contains a general discussion on how research is and can be performed within Enterprise Architecture. The second subsection covers the particularities of how the present work has been conducted and the research setting for each paper is considered individually.

2.1 RESEARCH WITHIN ENTERPRISE ARCHITECTURE

In a science of an artificially made world (both software and organizations – the two main categories of entities in EA – are obviously man-made), the rules of engagement differs somewhat from natural sciences since no infallible laws of nature exist (Simon 1996). At least they are too complicated to deduce. So, in order to hold a discussion on the research topic in this section, the base used here is found within the social science oriented work of King et al. (1994). In alignment to that work this section is divided into the areas of research objective, theory, research design, and data collection respectively.

2.1.1 RESEARCH OBJECTIVE

Choosing exactly what to research is not an altogether rational decision. Formulating a research question is arguably the most speculative part of research. According to King et al. (1994) there are however at least two principles that should guide this process. Firstly, "*a research project should pose a question that is "important" in the real world.*" Secondly, "a *research project should make a specific contribution to an identifiable scholarly literature …*" Even though these are long term goals for research, further reaching than a single Ph. D. project, these principles should guide the overall goals of the work. The former one addresses Enterprise Architecture as a discipline, and will not be argued for here. The importance of EA is left tacitly understood! The latter principle will be addressed per contributing paper in section 2.2.

2.1.2 THEORY

Formulating a useful theory is the heart of any scientific endeavor and the role of theory in Enterprise Architecture has also been a main theme through out this research. Oftentimes theory formulation is presented as the first step in research, but it need not to be since we cannot develop a theory without knowledge of prior work and some data of the real-world aspect of interest (King et al. 1994). In engineering disciplines (such as Enterprise Architecture), this problem becomes very obvious since the engineering can influence the scientific theories and vice versa, c.f. Figure 1 by Lawson (2004). Today, we live in a world of little science and much engineering, and harmonizing the two is not a simple task. In an artificial world, the laws of "nature" vary with the engineering manipulating it. So, in a volatile world where engineers are unable to let things alone, scientists clearly have a hard time. However, exactly how unstable the world of EA is will remain unspoken here.

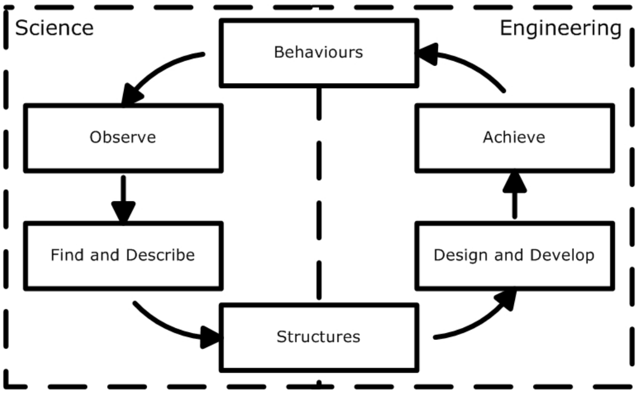

Figure 1. Goals of Science and Engineering by Lawson (2004).

When it comes to the usefulness of a theory King et al. (1994) proposes three principles according to which it could be evaluated. Firstly, Popper's principle of falsifiability (1980) should be followed. Concretely this means that, before the theory is tested, real-world observations that will prove the theory wrong should be identified. Secondly, in line with this first principle, theories with more observable implications, hence theories that are more testable, should be preferred over other theories. Thirdly, the more precise and concrete the theory states its predictions, the better.

EA is a discipline of multiple theories. We are far from being able to set up one theory for it. Instead, this thesis suggests that the discipline start using existing theories from other (sub-) disciplines such as IT security (cf. paper C), IT system maintenance, business and IT alignment, and knowledge management, etc. It is not the responsibility of EA to develop such theories, rather, in the long run, EA research should focus at finding techniques to combine these theories in a rational manner. Harshly, then, the mission of EA is not to develop new knowledge just reuse old findings in a new manner. This might seem dull or meaningless, but in fact it is an honorable endeavor motivated by explanations as the one given by Herbert Simon when he is elaborating on simulation as a method for producing 'new' knowledge: "All correct reasoning is a grand system of tautologies, but only God can make correct use of that fact. The rest of us must painstakingly and fallibly tease out the consequences of our assumptions." (1996). Apart from this challenge, a big difficulty often lies in how to operationalize or measure the employed theories. Even though arguably not a core occupation for EA, in practice the reused theories far from always will have suitable operationalizations attached.

Furthermore, the ideal of parsimony, i.e. keeping the theories simple, is considered important in the context of theory. King et al. (1994) argue that parsimony is not a principle that should be strived for regardless, rather it depends on the reality the theory is portraying; a complex reality simply cannot be parsimoniously described. In the field of EA, consensus reigns about its reality being a complex one. So should the principle of parsimony be abandoned? If it is only a statement about the real world it seems appropriate. However, the need for simple theories is urgent and the price to pay is credibility. This thesis argues for indicative theories favorably stating facts simple but with a non bulletproof reliability. The logic followed is that it is better to have a somewhat unreliable theory that is usable, than having a perfect theory that is to complex to use. The aim is to maximize leverage, i.e. to explain as much as possible with as little as possible, but the amount of input could be seen as more or less fixed.

Another aspect of leverage is the domain of validity for the theories. The justifying hypothesis of this research is that enterprise system management within one company in the electric power industry (in Sweden) could be generalized to other power companies (all over the globe) and, most importantly, to companies in many other industrial domains. Of course, it is the methods of enterprise architecting that we hope to reuse primarily, not the particular enterprise system solutions of the companies. This is already the prevailing conviction among the research community today; we find numerous books and articles with titles such as "Management of Information Technology" (Frenzel 1996), "Enterprise Application Integration" (Linthicum 2000), and "Building Systems from Commercial Components" (Wallnau et al. 2002); few of them are restricted to an industrial domain. Moreover, the electric power industry probably possesses one of the most multi-faceted IT milieus there are, wherefore it has served as an excellent empirical base for EA research. (cf. section 1.1)

2.1.3 RESEARCH STRATEGY

Research strategies or methods are the way in which theories are connected to the underlying data. In analogy, they can be seen as how the researcher chooses to argue for the hypothesis of the theory. There are many strategies present within science today, e.g. laboratory experiments, case studies, action research, surveys, archival analysis, and theorem proof includes a few (Yin 1994) (Galliers 1992). According to Yin (1994), the choice of research strategy is dependent on three conditions for the phenomenon that are to be examined: the type of question posed ("who," "what," "why," "how," etc.); the extent of control that the researcher has over the examined reality; and the degree of focus the study puts on contemporary respectively historical phenomena. When it comes to EA, probably most strategies are possible. In general, however, the biggest restriction is the sparse control over the environment, which typically rules out experiments and points towards case studies (Yin 1994).

2.1.4 Data Collection

The choice of research strategy also influences the way investigators collect data[5], i.e. observe the world. Altogether there are many different methods of observation; for instance interviews, participatory observation, inquiries, documentation review or use of measuring instruments. Each research strategy may combine these data collection methods as appropriate, but in general some are more common combinations. Case studies typically use interviews perhaps in combination with documentation review.

Quality of research design and data collection is often evaluated according to their reliability and validity (Yin 1994) (King et al. 1994). High reliability is achieved when the same procedure, applied several times to the same prerequisites, always produces the same results. In other words there are no stochastic errors apparent. Validity is the extent to which we actually measure what we think we are measuring. From a theoretical point of view, this measure is of uttermost importance when dealing with indicators (as is proposed for EA in this thesis). What constitute good indicators for different aspects of enterprise systems; for instance, how do we go about measuring IT security? When it comes to data collection validity is instead, for instance, a problem of whether interview respondents say and mean the same thing. Another crucial aspect of performing high quality research is to do it transparently. King et al. (1994) emphasize the importance of record and report the process with which data is collected so that the data generation (and the research design as a whole) could be replicated.

2.2 Design of Present Research

As illustrated, the present research has been quite theoretical in nature. In a world of a modest amount of science this might seem questionable. Oftentimes when this is the situation in a discipline, focus is put on explorative studies of how the real-world of interest is conditioned and is behaving. Rather than being empirically explorative, this research can be seen as theoretically explorative. Instead of *identifying* 'new' phenomena the objects of the real-world have to a large extent been considered as 'known' (even though this empirical journey has of course been done personally.) The only empirical unit of observation that can be considered 'new' within the research would be the aspect of information search cost. Of course, search cost has been a practical issue since long, but for some reason it has

[5] *Data* is here used to refer to information about the elements of the real world, as opposed to other parts of this thesis where the term refer to information kept within IT systems.

been much overlooked within EA. Consequently this topic, as well as other aspects of the research, has focused on framing the prerequisites of enterprise systems in theory. In its very ambitious version, this thesis aims at putting EA research on a more rigid footing.

Below the general settings for the present research is firstly presented. Secondly, the research settings for each included paper described. (Note that in this chapter only the methodological aspects of the papers are elaborated. A summary of the contribution of each paper is presented in chapter 3.)

2.2.1 GENERAL SETTINGS

When plunging into the world of EA, what is the unit of observation for enterprise systems? As said, we already know that there exist many interesting units, including; functionality, information (or data), IT-systems, security policies, knowledge, business processes etc. But one big challenge is to identify the most relevant units among all the possible ones (in order not to confuse the limitedly rational CIO). How the chains of causality are oriented within the discipline is not evident either. What are for instance the most important indicators for alignment of business and IT? The hypothesis of the present work is that the CIO concerns (elaborated in section 1.1.2) are the dependent variables and observation units like the above are the independent ditto. And connecting the independent to the dependent variables is the work of a theory. Paper C for instance examines how the dependent variable IT-security could be measured according to two rival theories.

From a theoretical perspective, the mission has thus been, and continues to be, to identify applicable existing theories that could be combined and adopted to EA. Except for the EA literature as such, disciplines that have been under consideration includes foremost Software Engineering in general and Software Architecture in particular, but also Systems Engineering, Information Systems Management, Business Modeling, and Decision Making and Behavior (cf. section 1.2.2).

Empirically, the electric power industry has been the information base for the research. Arguably it is one of the best industrial domains that can be chosen for EA research, since power companies possess highly complex enterprise systems in general. The reasons for this are many, a few are: The industrial processes are complex and was among the first to become automated; historically the electric power industry has invested heavily in IT and has in several aspects been in the technical front-line; the electricity infrastructure is truly national necessity, which allows issues security and safety to cost; moreover, national regulations have had direct influence on IT when the Swedish electricity market was deregulated in January 1, 1996; and since then, several mergers and acquisition have also taken place on the market, which of course have influenced the companies' enterprise systems heavily. It is the author's conviction that if choosing only one, the electric power industry is at least as beneficial as any other domain from a academic point of view.

Much of the empirical material presented in this thesis is derived from cooperation between the department of Industrial Information and Control Systems and the North European power company Vattenfall. The collaboration, known as the "CIO Think Tank", constitutes an in-depth long-term initiative where parts of the CIO group at Vattenfall and the researchers at the department regularly have working meetings and initiates case studies. The meetings as such can be seen as a distilled version of an action research setting (Galliers 1992), since the topics cover the present problems of the CIO. From this background, case studies on particular topics are started. Paper D is the outcome of such a case study initiative. From an academic point of view, the rationale behind this cooperation is to have a good understanding about the conditions in a particular industrial environment thus increase research efficiency. In the terminology of King et al. (1994), the CIO group helps the researchers with engaging in the most important research questions. Due to the complexity of enterprise systems, much effort must regularly be spent only on gaining background knowledge for each particular case study that is conducted. This effort can with this setting be decreased considerably. Furthermore, the CIO Think Tank cooperation enables the creation of what Yin (1994) labels a case study database, in which cohesive case study material can be stored for later reuse. (Finally, but not least importantly, this modus operandi is also a way of decreasing the time span from research findings to their industrial applications.)

This sub section has described general environment for this research project. Below the each paper is examined with respect to its research design and according to the structure of section 2.1. (The actual content of each paper is thus not summarized until the next chapter.)

2.2.2 Paper A

Paper A presents a research design that lies somewhere in between a laboratory experiment and theorem proofing (depending on the status of theory within software engineering.) The paper describes how formal Software Architecture theory is improper in the (real-world) context where deduction-based reasoning is practically impossible. Within Software Engineering (and Enterprise Architecture), "laboratory experiments" are often criticized for oversimplifying the aspects of real-world to such a degree that they are of doubted value. It is also the belief of the author that such criticism most often is justified. Consequently, the line of argumentation is somewhat reversed in comparison to traditional experiment setups; the article argue that even in the very small and (over) simplified experiment presented, the analyses proposed by (formalist) theory are filled with flaws. And it will not be more appropriate in more real-world like situations. Hence, the logic of the experiment is that the more critique that can be ascribed to the experiment, the more critique resides on the examined theory (given that the pinpointed analyses incredibility of the theory are true.)

In summary, the paper makes a contribution to existing literature in the sense that it illustrates that theories taking a formal approach is unsuitable for the present EA purposes.

2.2.3 PAPER B

In many senses paper B constitute a position paper of the outlook on EA research and practice held at the department of Industrial Information and Control Systems. It is based on a theoretical review (given the empirical situation in the electric power industry) and it criticizes the ad-hocness of present EA research. As an alternative, the paper suggests an approach outlining EA research and practice activities respectively. In particular, thoughts from decision theory (about the cost and utility of decisions) are called attention to and it is argued that this concept should be incorporated into the discipline of EA.

In summary, the paper examines the theoretical prerequisites and makes a contribution by re-evaluating (implicit) assumptions of EA.

2.2.4 PAPER C

Also Paper C is theoretical in nature. This paper is a direct continuation of Paper B, and it goes on examining how theories from other disciplines can be incorporated and used within EA. Two theories for assessing IT security are used as an example. Cases with fabricated (but realistic) data are used for evaluating the example theories used for EA purposes. Even though the data was not explicitly taken from a real-world setting or from a case reported in literature, it is an amalgamation of many (unreferenced) sources and is very standard, thus credible. And even if the data would turn out being controversial for some unforeseen reason, its complete correctness is not crucial to make the main contribution in the paper, since it focuses on the process of how theories can be incorporated and used within EA.

In summary, the paper describes how theory from other disciplines can be consistently incorporated and contribute to EA practice.

2.2.5 PAPER D

Paper D is based on a case study. It maps out how information about the enterprise system is scattered and communicated within the company. The study was of descriptive and explorative character. No pre-existing theory was evaluated or applied. The hypothesis was of very general kind and was merely used as a rationale for engaging in the case in the first place. (In principle the suspicion was that information used for CIO decision-making is often not up to date and that different information sources claim different things which makes it difficult to get hold of correct information.) The paper argues that these presumptions are important aspects that are widely disregarded in EA practice and research today. From the collected data some indicative common sense analyses (i.e. theory) were developed[6].

[6] For the observant reader it can be identified that this paper actually does what papers B and C are arguing against; it starts by developing a model without making sure how important this aspect is to the CIO, or identify-

The data collection consisted of interviews and document reviews. A convenient condition in this research setting was that the aim was to imitate the information that the CIO group would have gotten if they had done this particular exercise (say, before a major project or so). Thus, to a large extent the problem of validity boils down to whether or not respondents would give the same answers to a researcher as to the CIO (or their staff). In the way the case study was conduced some difficulties were encountered regarding the terminology due to the fact that the academic and practical lingo differed. Whether or not respondents actually were telling the 'truth' about how the real-world is actually constituted (regarding the enterprise system) is interestingly enough of minor significance here. Firstly, since the problem is the same for the CIO, but also since as far as the CIO is concerned the 'truth' becomes for practical reasons the sum of all respondents and documents addressed. Still for other used measures, validity is still important. For instance, are the documents correctly interpreted and did respondents answer truthfully about how they spread and receive information?

In summary, Paper D contributes by explicitly suggesting how the largely neglected EA aspect of information distribution and organization could be modeled.

2.2.6 Paper E

Paper E makes an excursion into the heavily connoted area of complexity. The investigation is done from the point of view of the limitations of the human mind. The overall logic of the paper is that models (used for decision support) can express information (semantics) more or less appropriate for human minds. In the paper a syntactic complexity measure is applied on the models, as a very rough substitute for a good cognitive derived measure. Four models (two state-of-the-art and two best-practice) from the discipline of project management are used as examples on which this measure of easiness of information elicitation (called "elegance" in the paper). Consequently, empirically speaking this paper includes another experiment or simulation of where models are used for exemplifying application of theory. The validity of the theory (measure) is suggested to be broader than covering only a project management perspective, basically it can be applied to any architecturally oriented model, wherefore it suits well with Enterprise Architecture where this aspect is commonly overlooked.

In summary, this paper contributes with a theory or analysis procedure for assessing "elegance" of different types of models (i.e. notations).

ing what existing theory that speaks of this issue. The theoretical, but perhaps weak, response to such observation would be that it is obvious for each and everyone that this is an important matter, and further there are no theories for it (at least none appropriate was found), so consequently new ones must be developed. And as King et al. (1994) points out, data collection and theory development is an iterative process in which sometimes heavy data collection initiates the course of action. A pragmatic response would be that this case study was conducted before papers B and C were written and it easy to be seduced by the EA guise!

Chapter 3

Summaries of Included Papers

This chapter introduces the papers included in the thesis. Regarding the indication of the different authors' contributions to the respective paper, it is specified with one out of two principles. When the names are ordered alphabetically, an equally shared contribution is indicated (Papers A and E), otherwise the order of appearance denote the amount of contribution (Papers B, C, and D).

PAPER A

Exploring Architectural Analysis Credibility from a Developer Perspective

Mathias Ekstedt and Pontus Johnson

In the Proceedings of the Fourth Australasian Workshop on Software and Systems Architecture, Sydney, Australia, 2002.

This paper explores the credibility of deduction-based analysis methods for software architecture. Early assessment of system characteristics in software development is one of the main concerns of the discipline of software architecture. A disadvantageous architecture may introduce several undesired properties to the system, requiring expensive corrections later in the project. In an attempt to mitigate this risk, several analysis methods have been proposed for a significant set of system properties such as performance, modifiability, reliability, and security. These methods range from a formal basis to qualitative reasoning about the attributes. This paper focuses on circumstances that may corrupt analyses of software architectures, in particular a number of issues related to transformations between specifications. The transformation of one kind of software specification into another is at heart of software engineering. Software developers transform requirements specifications into architectural specifications into design specifications and so on, until the system is eventually implemented.

Recognizing the limited success of formal methods for program transformation and architectural refinement in an information-laden world of enterprise-level of Software Architecture (assuming complete correctness in transformations), the paper considers how informally devised transformations between specification languages may invalidate the results of architectural analyses. An example with a simple system transformed between the languages of Wright, UML, and C++, considers a number of transformation distortions and their effects for analysis reliability. Unless a formal approach can be taken throughout the development of the software, its functionality and qualities can not be assessed with complete reliability from architecture models.

If a deduction-based approach to software development is to be kept and formal methods cannot be assumed, consequences of the findings include a need for harmonization of software abstractions used in the same development project, including component technologies, programming languages and modeling languages. Component certification is suggested as a means for increasing the trust in the architectural specification, and thereby the architectural analysis. Another suggested means for increasing the trustworthiness of the architectural analysis is usage of (certified) automated transformers, such as code generators or compilers. Finally, assessment of the input data sensitivity of architectural analysis methods is suggested as a way to increase the credibility of the analysis results.

PAPER B

Consistent Enterprise Software System Architecture for the CIO: A Utility-Cost Based Approach

Mathias Ekstedt, Pontus Johnson, Åsa Lindström, Magnus Gammelgård, Erik Johansson, Leonel Plazaola, Enrique Silva

In the proceedings of the 37th Hawaii International Conference on System Sciences, Big Island, Hawaii, U.S.A., 2004.

This paper presents an approach for conducting an Enterprise Architecture initiative. The approach takes its starting point in the needs of the Chief Information Officer (CIO) and presents a set of activities to be carried out. The reasons for promoting the approach are three, so far neglected, aspects of Enterprise Architecting. Firstly, Enterprise Architecture models have forced collection of much information about the world without knowing precisely what conclusions to draw from it. This logic should be reversed. Secondly, there are too many points on the CIO's agenda to engage in them all. The assignment list must thus be traded off between the utility of completing the tasks on the list with the cost of collecting information for doing so. Thirdly, Enterprise Architecture frameworks do not ensure consistency among its models (views).

In short, the first activity of the approach suggests that the CIO come up with a list of set of questions or areas of concern (e.g. security, governance, alignment, and cost) prioritized in order of importance given the company situation. Given a set of analysis criteria (i.e. theory) for assessing each question, it can be determined what information that needs to be collected. An estimation of about the cost of answering the questions is then conducted. A utility-cost trade-off then motivates a re-prioritization among the questions. Not until this point are any models developed. Before models are instantiated, a (company-specific) meta-model is developed. Given this method of attacking Enterprise Architecture endeavors, requirements are also put on the support that is needed by CIO. For instance would a standard set of important questions and a general meta-model be of much help, as would methods for estimating utility, search cost, and utility cost trade-offs. Finally, of course developing the analysis criteria is not an assignment suitable for the CIO. Rather, all those issues constitute an agenda for the research community.

Paper C

Using Enterprise Architecture for CIO Decision-Making: on the Importance of Theory

Pontus Johnson, Mathias Ekstedt, Enrique Silva and Leonel Plazaola

In Proceedings of the Second Annual Conference on Systems Engineering Research, Los Angeles, U.S.A., 2004.

For architectural models to function as decision-making support, this paper argues that they must be amenable to architectural analysis. The paper demonstrates the importance of architectural theory in the analysis of Enterprise Architecture models of the enterprise system. The concept of a*rchitectural theory diagrams* are proposed as means for presenting architectural theories. The diagram serves three main purposes. Firstly, it can be used to compare competing theories to each other. By these comparisons, the theory that best suits our purposes can then consequently be chosen. Secondly, it will detail what information a good architectural model needs to contain. Thirdly, it makes the theory explicit, facilitating both critical examination and reuse of the theory.

An included example is based on two competing state-of-the-art theories for IT security. In addition, two enterprise system scenarios are provided for indicating different potential futures of a company. The evaluation of the scenarios indicates different results (i.e. different level of IT security) for the two theories. Consequently, it becomes important to evaluate theories. Three aspects are promoted for architectural theory quality; relevance (to the CIO), credibility, and information search cost. Finally, a meta-model aligned with one of the example theories is outlined.

PAPER D

The Architectural Information View for the Power Electricity Industry

Mathias Ekstedt, Pontus Johnson, and Peter Sjölin

In Proceedings of CIGRÉ SC D2 Colloquium, Rio de Janeiro, Brazil, 2003.

This paper proposes a new view (or viewpoint) for the Enterprise Architecture; the Architectural Information View. It is derived from the need of further abstract away system details at the enterprise level of systems. The view does not speak of systems as such; instead it focuses on where and how system architectural information and knowledge is allocated within the organization. The center of attention is thus put on information carriers, constituted by various forms of documentation as well as people possessing architecturally significant information. In the view, the locations of, awareness of, information exchanged among, and the contents of architectural information carriers are represented. The article argues that the view is central for performing analyses of organizational information structures related to the enterprise system; e.g. identifying information inconsistencies, inefficient information flows, incomplete architectural awareness, as well as identifying central persons and documents. The view is proposed both as an attempt to find a higher level abstraction of systems, as well as an attempt to model a highly neglected area within Enterprise Architecture; the cost of searching for information needed for the models of Enterprise Architecture frameworks.

The Architectural Information View is illustrated with a case study performed at the North European energy company Vattenfall. One of the main objectives with the study was to examine how the CIO group at Vattenfall on average is informed about the parts of their enterprise system. The study was delimited to architectural information about one specific software system, representing an arbitrary system, out of the approximately 350 IT-systems large enterprise system. In addition to the CIO group, the study was organization-wise primarily focused on the business unit being in the owners and having the maintenance responsibility of the system under examination.

PAPER E

Making Project Complexity Understandable: the Elegance of Notations

Mathias Ekstedt, Pontus Johnson, Joakim Lillieskold and Narcisa Jonsson

In Proceedings of the 12th International Conference on Management of Technology, Nancy, France, 2003.

Models are the prime instrument for managing complexity in Enterprise Architecture, as well as in most other scientific disciplines. This paper explores architectural models' capability of managing complexity when respect is given to the limitations of the human mind. The relation between consumers and their models is a multi-facetted problem domain, and this paper is delimited to two aspects of this; *expressiveness* and *ease of use* of the models. Expressiveness illustrates that a model must be able to speak of the right set of questions, namely the ones preferred by the model consumer. This aspect is thus closely related to the concept of views. Models' ease of use relate to how intuitively the information is presented. Using a world map as an example, areas of countries are certainly an issue that is expressed therein. However, the ease of use of eliciting that information from the model depends on what type of geometrical projection that is used, and in many cases would a list of numerical figures certainly be preferable to the map for that analysis. The paper introduces the concept of elegance for measuring the relation between expressiveness and ease of use of models (and meta-models). More formally the elegance of a particular language is defined in the paper as the relationship between its semantic utility and its syntactic complexity.

An included example evaluates elegance of two state-of-the-art and two best-practice project dependency languages, with respect to the needs of a project planner. From the perspective of Enterprise Architecture, the choice of illustration is not an obvious or optimal one, even though far from irrelevant. The reason behind this choice was that the paper was a cooperation among several authors, and actual the outcome of the elegance analyses conducted was of particular interest to one of the other authors. The four sample notations however serve the purpose of illustrating the general concept elegance well, which is the main rationale in relation to this thesis. In the highly model-based approach of Enterprise Architecture for managing complexity, elegance is a measure that should guide pragmatism of (meta-) models.

RELATED PUBLICATIONS NOT INCLUDED IN THE THESIS

A Survey on CIO Concerns - Do Enterprise Architecture Frameworks Support Them?

Åsa Lindström, Pontus Johnson, Erik Johansson, Mathias. Ekstedt, and Mårten. Simonsson

Submitted to Journal of Information Systems Frontiers, Kluwer Academic Publichers. (2004-03-01)

Management of Enterprise Software System Architectures: Focusing on Information Economy and Model Consistency

Mathias Ekstedt, Pontus Johnson, Åsa Lindström, Erik Johansson, Lars Nordström

Proceedings of the Third Conference on Software Engineering Research and Practice in Sweden, SERPS'03, Lund, Sweden, 2003.

Managing Complex IT-Projects – A Need for a Tool Addressing Technical and Organizational Complexity

Joakim Lilliesköld and Mathias Ekstedt

Presented at the 17th Nordic Conference on Business Studies (NFF), Reykavik, Iceland, 2003.

Management of Enterprise Information Systems for Power System Control and Operation

Torsten Cegrell, Mathias Ekstedt, and Patrik Forsgren

In Proceedings of the 5th International Conference on Power System Management and Control, London, United Kingdom, 2002.

On Software Models

Mathias Ekstedt

External Report, Industrial Information and Control Systems, Royal Institute of Technology (KTH), 2001.

Chapter 4
Further Works

In a discipline so young and with such an extensive scope as Enterprise Architecture, there are of course a vast set of further works that have to be conducted. And when the options are so many it seems logical that research within the discipline should follow the same principle as the one being promoted in this thesis; namely focus on questions with expected high utility and low cost for finding the corresponding answer. Having that said, finding *that* list has, however, not been the aim of this research. Yet, to the author, there are three areas that seem to be reasonable for further exploration in order to make Enterprise Architecture even more practically useful in the future:

Tools. Architectural models are at heart of EA, and consequently they must not fail its practitioners. Using a carefully designed modeling tool is thus probably uncontroversial. For instance, the above pin-pointed topic of model (or viewpoint) consistency seems practically impossible to handle without computerized support. There are today a fairly extensive supply of vendors selling such software products, cf. IFEAD (2004). As argued previously however, models fall short if they are not guided by purpose and theory. How theories are to be incorporated and analyses automated is still to be elaborated.

Information search cost. The cost of gathering EA relevant information from the company can be extremely high. In order not to be yet-another-academically-elegant approach which is practically not viable, the information search cost (plus the implementation cost) cannot exceed the utilities. A great challenge is thus to obtain good leverage on EA. Consequently, a good estimation method of information collection cost is vital.

Reuse of existing theories. Since EA is a broad and multi-facetted discipline resting on top of other disciplines, it would be unwise not to reuse existing knowledge (e.g. from areas such as software security, software maintenance, software architecture, business process modeling, and knowledge management). The main task for EA is to define a solid base for determining what already existing knowledge that is suitable, or perhaps how it could be adapted, and how they can be combined. Probably the most obvious challenge in adapting theories is to identify an appropriate level of detail on which no further information will be taken into account.

Chapter 5
Concluding Remarks

Enterprise Architecture (EA) is a recipe for management of an organization's IT related issues, including several non-technical aspects. Today however, the ingredients are far from well defined; we do know some main elements, but the proportions remain unclear and the more fine-tuned features are still not added. This thesis depicts EA incorporating at least some aspects of Software Engineering, Software Architecture, Information Systems, Organizational Theory, and Decision Making. Using an analogy one might say that it is of course possible to fill our EA stew with all kinds of flavors, but will it taste better? Too many different flavors will certainly challenge each other, and the just-right mixture of ingredients is determined by the taste of the gourmand. EA must thus define the scope of its sub-disciplines given a perspective of some stakeholder, which this thesis argues is the Chief Information Officer (CIO). One might object and argue that CIOs have different interests and assignments in different organizations. But in order to justify the existence of EA we must be able to generalize all those CIOs into one (or perhaps a set of) stereotype(s). And this thesis argues that it is that aggregated need should drive EA research and development. Continuing with the analogy, this thesis promotes one particular fine-tuned EA spice that is seldom included in the stew at the moment; namely the aspect of enterprise system knowledge and information distribution in the company.

Furthermore, instructions for how to prepare the stew so that the taste will be the desired one (i.e. the one of the CIO) are lacking. Firstly, it is not commonly articulated what the CIO's preferences are when EA initiatives are started, and secondly, when it comes to the instructions, i.e. analysis procedures, or theories, the lack of these obstructs achievement of an effective discipline. This thesis tries to motivate and illustrate the benefit of the use of theory in a discipline where there currently is much practice and hardly any (explicit) science. It is however probably in the distant future before EA primarily is about optimizing the reality according one or a couple of theories (such as when a mechanical engineer is using aerodynamics theory for optimizing the trade-off between speed and maneuverability of aeroplanes; even though this task is a great challenge with many experiments and simulations, the theoretical prerequisites are set.) Rather, only identifying prospects for such theories are assignments enough for EA for quite some time ahead. And just as recipes for a stew, the theories for EA are expected to be of an indicative and heuristic nature.

The saying claim that there is no free lunch, and also the ingredients of our EA stew are costly to acquire. Sometimes the price paid even exceeds the pleasure of the taste. I.e., collecting information for developing an Enterprise Architecture model can be very expensive. Today this aspect is a highly neglected. This thesis argues this price has to be traded off against the model's utility in terms of analyses it can provide. For maintenance and practical reasons, it is also important that the chosen EA models are kept consistent. Unfortunately, the analogy finally fails for illustrating that different models are more or less easy for the CIO to analyze depending on what language that is used due to the fact that even though all the necessary information might be in the model, all humans suffer from limited mental capabilities.

A theme throughout this thesis is a search for academic rigor for Enterprise Architecture. Even though perhaps some pessimistic phrases are found herein, the future for the discipline looks bright. Much attention and work is directed to the subject, that is, both from practitioners, but also increasingly from academia. And given that Enterprise Architecture to a large extent does not have to come up with new knowledge, just re-package existing; the potential progress of the discipline is promising.

References

Alagar, V., K. Periyasamy (1998), *Specification of Software Systems*, Springer, 1998

Allen, R. (1997), *A Formal Approach to Software Architecture*, Ph.D. Thesis, Carnegie Mellon University, 1997

Andersson, J. (2002), *Enterprise Information Systems Management: An Engineering Perspective on the Aspects of Time and Modifiability*, Ph.D. Thesis, Royal Institute of Technology (KTH), 2002.

Armour, F. and S. Kaisler (2001), "Enterprise Architecture: Agile Transition and Implementation," *IEEE IT Professional Vol. 3, No. 6*, 2001.

Armour, F., S. Kaisler, and J. Getter (2002), "A UML-driven Enterprise Architecture Case Study," *Proceedings of the 36th Hawaii International Conference on System Sciences*, 2002.

Armour, F., S. Kaisler, and S. Liu (1999a), "A Big-Picture Look at Enterprise Architecture", *IEEE IT Professional Vol. 1, No 1*, 1999.

Armour, F., S. Kaisler, and S. Liu (1999b), "Building an Enterprise Architecture Step by Step", *IEEE IT Professional Vol. 1, No 4*, 1999.

Axelsson K. and G. Goldkuhl (1998), *Strukturering av informationssytem: arkitekturstrategier I teori och praktik*, Studentlitteratur, 1998. (In Swedish)

Bachmann, F., L. Bass, G. Chastek, P. Donohue, and F. Peruzzi (2000), *The Architecture-Based Design Method*, Technical Report CMU/SEI-2000-TR-01, 2000.

Baragry, J. and K. Reed (1998), "Why Is It So Hard To Define Software Architecture?," *Proceedings of the Asia Pacific Software Engineering Conference*, 1998.

Barbacci, M., M. Klein, and C. Weinstock (1997), Principles for Evaluating the Quality Attributes of a Software Architecture, Technical Report CMU/SEI-96-TR-036, 1997.

Barbacci, M., M. Klein, T. Longstaff, and C. Weinstock (1995), *Quality Attributes*, Technical Report CMU/SEI-95-TR-021,1995.

Bass, L., P. Clements, and R. Kazman (1998), *Software Architecture in Practice*, Addison-Wesley,1998

Beck, K. (2000), *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 2000.

Berczuk, S., and B. Appleton (2003), *Software Configuration Management Patterns: Effective Teamwork, Practical Integration*, Addison-Wesley, 2003.

Boar, B. (1993), *The Art of Strategic Planning for Information Technology: Crafting Strategy for the 90s*, John Wiley & Sons, 1993.

Boar, B. (1999), *Constructing Blueprints for Enterprise IT Architectures*, John Wiley & Sons, 1999.

Boehm, B. (1988), "A Spiral Model of Software Development and Enhancement," *Computer Vol. 21, No. 5*, 1988.

Boehm, B., J. Brown, H. Kaspar, M. Lipow. G. MacLeod, and M. Merritt (1978), *Characteristics of Software Quality*, North-Holland Publishing, 1978.

Booch, G. (1991), *Object Oriented Design with Applications*, Benjamin/Cummings, 1991

Boster, M., S. Liu, and R. Thomas (2000), "Getting the Most from Your Enterprise Architecture," *IEEE IT Professional Vol. 2, No. 4*, 2000.

Briand, L., J. Daly, and J. Wüst (1999), "A Unified Framework for Coupling Measurement in Object-Oriented Systems," *IEEE Transaction on Software Engineering, Vol. 25, No. 1*, 1999.

Britton, C. (2001), *IT Architectures and Middleware: Strategies for Building Large, Integrated Systems*, Addison-Wesley, 2001.

Brooks, F. (1995), *The Mythical Man-Month, Anniversary Edition: Essays on Software Engineering*, Addison-Wesley, 1st ed. 1975, 1995.

Brown A. (2000), *Large-Scale, Component-Based Development*, Prentice-Hall, 2000

Brown, A. and B. Barn (1999), "Enterprise-Scale CBD: Building Complex Computer Systems from Components," *Proceedings of Software Technology and Engineering Practice*, 1999.

Brown, W., R. Malveau, H. McCormick III, and T. Mowbray (1998), *AntiPatterns: Refractoring Software, Architectures, and Project in Crisis*, John Wiley & Sons, 1998.

Bruzelius, L., and P-H. Skärvad (1995), *Integrerad organisationslära*, Studentlitteratur, 1995. (In Swedish.)

Buschmann, F., R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal (1996), *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*, Wiley, 1996.

Checkland, P. (1981), *Systems Thinking, Systems Practice*, John Wiley & Sons, 1981.

CIMOSA, The CIMOSA Association web site (2004), *http://www.cimosa.de* , accessed September 7, 2004.

CIO Council (1999), *The Federal Enterprise Architecture Framework Version 1.1*, Chief Information Officer Council, 1999.

CIO Council website (2004), *http://cio.gov*, accessed April 1 2004

Clements, P. and L. Northrop (1996), *Software Architecture: An Executive Overview*, Technical Report CMU/SEI 96-TR-003, 1996.

Clements, P., F Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Stafford (2003), *Documenting Software Architectures: Views and Beyond*, Addison Wesley, 2003.

Clements, P., R. Kazman, and M. Klein (2002), *Evaluating Software Architectures: Methods and Case Studies*, Addison-Wesley, 2002.

Clinger-Cohen Act (1996), Division E, National Defense Authorization Act for FY, P.L. 104-106, February 10, 1996.

Crnkovic, I. and M. Larsson (2002), *Building Reliable Component-Based Software Systems*, Artech House, 2002.

Davenport, T. (1993), *Process Innovation – Reengineering Work through Information Technology*, Harvard Business School Press, 1993.

Davies, A. (1993), *Software Requirements: Objects, Functions, and States*, Prentice Hall, 1993.

Denning, P. and P. Dargan (1994), "A Discipline of Software Architecture," *ACM Interactions, Vol. 1, No. 1*, 1994.

Dijkstra, E., *A Dicipline of Programming*, Prentice Hall, 1976.

DoD Architecture Framework Working Group (2003), *DoD Architecture Framework Version 1.0*, Department of Defense, 2003.

DoD C4ISR Architectures Working Group (1997), *C4ISR Architecture Framework Version 2.0*, Department of Defense, 1997

EAC, Enterprise Architecture Community (2004) *http://www.eacommunity.com*, accessed April 1, 2004

Emmerich, W., E. Ellmer, and H. Fieglein (2001), "TIGRA - An Architectural Style for Enterprise Application Integration," *Proceedings of the 23rd International Conference on Software Engineering*, 2001.

Eriksson, H-E. and M. Penker (2000), *Modeling business with UML: Business Patterns at Work*, OMG Press, 2000.

FEAPMO, Federal Enterprise Architecture Program Management Office (2003a), *The Performance Reference Model Version 1.0: A Standardized Approach to IT Performance*, 2003.

FEAPMO, Federal Enterprise Architecture Program Management Office (2003b), *The Business Reference Model Version 2.0: A Foundation for Government-wide Improvement*, 2003.

FEAPMO, Federal Enterprise Architecture Program Management Office (2003c), *The Service Component Reference Model (SRM) Version 1.0: A Foundation for Government-wide Improvement*, 2003.

FEAPMO, Federal Enterprise Architecture Program Management Office (2003d), *The Technical Reference Model (TRM) Version 1.1: A Foundation for Government-wide Improvement*, 2003.

FEAPMO, Federal Enterprise Architecture Program Management Office website (2004), *http://www.feapmo.gov*, accessed April 1, 2004.

Fenton, N. and M. Niel (1999), "Software Metrics: successes, failures, and new directions," *Elsevier Journal of Systems and Software, Vol. 47, No. 2-3*, 1999.

Fenton, N. and S. Pfleeger (1997), *Software Metrics: A Rigorous and Practical Approach, 2nd ed.*, PWS Publishing, 1997.

Frenzel, C. W. (1996), *Management of Information Technology*, 2nd ed., CTI, Course Technology, International Thomson Publishing, 1996.

Galliers, R. (1992), *Information Systems Research: Issues Methods and Practical Guidelines*, Blackwell Scientific Publications, 1992.

Gamma, E., R. Helm, P. Johnson, and J. Vlissides (1995), *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1996.

Garlan, D. (2000), "Software Architecture: a Roadmap," In A. Finkelstein (ed.), *The Future of Software Engineering: 22nd International Conference on Software Engineering*, 2000.

Garland, J. and R. Anthony (2003), *Large-Scale Software Architecture: A Practical Guide Using UML*, John Wiley & Sons, 2003.

GEAO, Global Enterprise Architecture Organization web site (2004), *http://www.geao.org*, accessed April 1, 2004

Gottschalk, P., and N. Taylor (2000), "Strategic Management of IS/IT Functions: The Role of the CIO," *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2000.

Haglind, M. (2002), *Information Systems Planning in the Deregulated Electric Power Industry: Addressing Small and Medium-sized Enterprises*, Ph.D. Thesis, Royal Institute of Technology (KTH), 2002

Hammer, M. and J. Champy (1994), *Reengineering the corporation: A Manifesto for Business Revolution*, Harper Business books, 1994.

Hastie, R. and R. Dawes (2001), *Rational Choice in an Uncertain World: The Psychology of Judgment and Decision Making*, Sage Publications, 2001.

Heineman, G. and W. Councill (Eds.) (2001), *Component-based software engineering: Putting the pieces together*, Addison-Wesley, 2001

Hoffmeister, C., R. Nord, and D. Soni (2000), *Applied Software Architecture*, Addison-Wesley, 2000.

IEEE Std 1471-2000 (2000), *IEEE Recommended Practice for Architectural Description of Software-Intensive Systems*, 2000.

IEEE SWEBOK (2001), *Guide to the Software Engineering Body of Knowledge, Trial Version*, IEEE Computer Society, 2001.

IFEAD, Institute for Enterprise Architecture Developments website (2004), *http://www.enterprise-architecture.info*, accessed April 1, 2004.

INCOSE, International Council on Systems Engineering (2000), *Systems Engineering Handbook: A "How To" Guide for all Engineers, Version 2.0*, International Council on Systems Engineering, 2000.

Jacobson, I., M. Christerson, P. Jonsson, and G. Övergaard (1992), *Object-Oriented Software Engineering: A Use Case Driven Approach*, ACM Press, Addison-Wesely, 1992.

Johnson, P. (2002), *Enterprise Software System Integration: An Architectural Perspective*, Ph.D. Thesis, Royal Institute of Technology (KTH), 2002.

Jonkers, H. et al. (2003), "Towards a Language for Coherent Enterprise Architecture Descriptions," *Proceedings of the 7th IEEE International Enterprise Distributed Object Computing Conference (EDOC'03)*, 2003.

King, G., R. Keohane, and S. Verba (1994), *Designing Social Inquiry: Scientific Inference in Qualitative Research*, Princeton University Press, 1994.

Kirkland, N. (2002), "CIO Agenda for Productivity and Growth", *Gartner Symposium ITxpo*, Cannes, France, 4-7 November, 2002.

Kirkpatrick, T. A. (2002), "Role of the CIO," *CIO Insight*, April 15 2002. (http://www.cioinsight.com/article2/0,1397,2310,00.asp)

Kobryn, C. (1998), "Modeling Enterprise Software Architectures Using UML," *Proceedings of Second International Enterprise Distributed Object Computing Workshop*, 1998.

Kotonya, G. and I. Sommerville (1998), *Requirements Engineering: Processes and Techniques*, John Wiley and Sons, 1998.

Kruchten, P. (1995), "The 4 + 1 View Model of Architecture," *IEEE Software, Vol. 12, No. 6*, 1995.

Kruchten, P. (2000), *The Rational Unified Process: An Introduction*, 2nd ed. Addison-Wesley, 2000.

Lawson, H. (2004), "Managing Complexity in a Learning Organization: Thinking and Acting in Terms of Systems," *Systems Engineering: The Journal of the International Council on Systems Engineering, In publication*, Wiley, (2004)

Leon, A. (2000), *A Guide to Software Configuration Management*, Artech House, 2000.

Linthicum, D (2000), *Enterprise Application Integration*, Addison Wesley, 2000.

Luftman, J. (ed.) (1996), *Competing in the Information Age: Strategic Alignment in Practice*, Oxford University Press, 1996.

Lutz, J. (2000), "EAI Architecture Patterns," *eAI Journal*, 2000.

Macaulay, L. (1996), *Requirements Engineering*, Springer, 1996.

Maier, M. and E. Rechtin (2000), *The Art of Systems Architecting*, 2nd ed., CRC Press, 2000.

March, J. (1994), *A Primer on Decision Making: How Decisions Happen*, Free Press, 1994.

Martin J. N. (1997), *Systems Engineering Guidebook: A Process for Developing Systems and Products*, CRC Press, 1997.

McGovern, J., S. Ambler, M. Stevens, J. Linn, E. Jo, and V. Sharan (2003), *A practical Guide to Enterprise Architecture*, Prentice Hall, 2003.

Medvidovic, N. and R. Taylor (2000), "A Classification and Comparison Framework for Software Architecture Description Languages," *IEEE Transaction on Software Engineering, Vol. 26, No. 1*, 2000.

Mehta, N., N. Medvidovic, and S. Phadke (2000), "Towards a Taxonomy of Software Connectors," *Proceedings of the International Conference on Software Engineering*, 2000.

Moorman Zaremski, A. and J.Wing (1997), "Specification Matching of Software Components", *ACM Transaction on Software Engineering and Methodology, Vol. 6, No. 4*, 1997.

Morabito, J., I. Sack, and A. Bhate (1999), *Organization Modeling: Innovative Architectures for the 21st century*, Prentice Hall, 1999

Moriconi, M. and X. Qian (1994), "Correctness and Composition of Software Architectures," *Proceedings of the 2nd ACM SIGSOFT symposium on Foundations of software engineering*, 1994.

Moriconi, M., X. Qian, and R. Riemenschneider (1995), "Correct Architecture Refinement," *IEEE Transaction on Software Engineering, Vol. 21, No. 4*, 1995.

Nilsson, A., C. Tolis, and C. Nellborn.(1999), *Perspectives on Business Modeling: Understanding and Changing Organizations*, Springer, 1999

O'Rourke, C., N. Fishman, and W. Selkow (2003), *Enterprise Architecture, Using the Zachman Framework*, Thomson Course Technology, 2003.

OMG, Object Management Group (2003), *OMG Unified Modeling Language Specification, Version 1.5*, Object Management Group, March 2003. (http://www.omg.org/docs/formal/03-03-01.pdf)

Oskarsson, Ö. (1982), *Mechanisms of Modifiability in Large Software Systems*, Ph. D. Thesis, Linköping University, Sweden, 1982.

Parnas, D. (1972), "On the Criteria To Be Used in Decomposing Systems into Modules," *Communications of the ACM, Vol. 15, No. 12*, 1972.

Perks, C., and T. Beveridge (2003), *Guide to Enterprise IT Architecture*, Springer Verlag, 2003

Perry, D. and A. Wolf (1992), "Foundations for the Study of Software Architecture," *ACM Software Engineering Notes, Vol. 17, No. 4*, 1992.

Popper K. (1980), *The Logic of Scientific Discovery*, Hutchinson, 1st impression 1959, 1980.

Pressman, R. and D. Ince (2000), *Software Engineering: A Practitioner's Approach*, European Adaptation, 5th ed., McGraw-Hill, 2000.

Ronin (2004), Ronin International Inc. Enterprise Unified Process website, *http://www.enterpriseunifiedprocess.info*, accessed April 1, 2004.

Royce, W. (1970), "Managing the Development of Large Software Systems," *Proceedings of the IEEE WESCON*, 1970.

Rubenstein, A. (1998), *Modeling Bounded Rationality*, MIT Press, 1998.

Ruh, W., F. Maginnis, and W. Brown (2001), *Enterprise Application Integration*, John Wiley & Sons, 2001.

Schmidt, D., M. Stal, H. Rohnert, and F. Buschmann (2000), *Pattern-Oriented Software Architecture, Volume 2: Patterns for Concurrent and Networked Objects*, Wiley, 2000.

Sessions, R. (2003), *Software Fortresses: Modeling Enterprise Architectures*, Addison-Wesley, 2003.

Shaw, M. (1989), "Larger Scale Systems Require Higher-Level Abstractions," *Proceedings of the 5th international workshop on Software specification and design*, 1989.

Shaw, M. and D. Garlan (1996), *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, 1996.

Simon, H. (1996), *The Science of the Artificial*, 3rd ed., MIT Press, 1996, 1st ed. 1969.

Simon, H. (1997), *Administrative Behavior: A Study of Decision-Making Processes in Administrative Organizations*, 4th ed. Free Press, 1997. 1st ed. 1947.

Simon, H. (1997b), *Models of Bounded Rationality: Volume 3, Empirically Grounded Economic Reason*, MIT Press, 1997.

Skinner, D. (1999), *Introduction to Decision Analysis: A Practitioner's Guide to Implementing Decision Quality*, 2nd ed. Probabilistic Publishing, 1999.

Sowa, J. and J Zachman (1992), "Extending and Formalizing the Framework for Information Systems Architecture," *IBM Systems Journal, Vol. 31, No 3*, 1992.

Spewak, S. (1992), *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*, John Wiley & Sons, 1992.

Spitznagel, B. and D. Garlan (1998), "Architecture-Based Performance Analysis," *Proceedings of the 1998 Conference on Software Engineering and Knowledge Engineering*, 1998.

Stevens, R., P. Brook, K. Jackson, and S. Arnold, *Systems Engineering: Cooping with Complexity*, Prentice Hall, 1998

Szyperski, C. (1998), *Component Software: Beyond Object-Oriented Programming*, Addison-Wesley, 1998.

Taylor, I. (1992), "A Process Reference Model for Large-Scale Software Development," *Proceedings of the Second International Conference on Systems Integration*, 1992.

TOG, The Open Group (2002), *The Open Group Architectural Framework Version 8*, The Open Group, 2002

Tyndale-Biscoe, S., O. Sims, B. Wood, and C. Sluman (2002), "Business Modeling for Component Systems with UML," *Proceedings of the 6th IEEE International Enterprise Distributed Object Computing Conference (EDOC'02)*, 2002

Wallnau, K., S. Hissam, and R. Seacord (2002), *Building Systems from Commercial Components*, Addison-Wesley, 2002.

Ward, J., and P. Griffiths (1996), *Strategic Planning for Information Systems*, 2nd ed., John Wiley and Sons, 1996.

Wegmann, A. and O. Preiss (2003), "MDA in Enterprise Architecture? The Living System Theory to the Rescue… ," *Proceedings of the 7th IEEE International Enterprise Distributed Object Computing Conference (EDOC'03)*, 2003.

Wing, J. (1990), "A Specifier's Introduction to Formal Methods," *IEEE Computer Vol. 23, No. 9*, 1990.

Wirth, N. (1974), "On the Composition of Well-Structured Programs," *ACM Computing Surveys*, Vol. 6, No. 4, 1974.

Yin, R. (1994), *Case Study Research: Design and Methods*, 2nd ed., Sage Publications, 1994.

Zachman, J. (1987), "A Framework for Information Systems Architecture," *IBM Systems Journal, Vol. 26, No 3*, 1987.

ZIFA (2004), The Zachman Institute for Framework Advancement website, http://www.zifa.com, accessed April 1, 2004.

Zuse H. (1997), *A Framework of Software Measurement*, Walter de Gruyter, 1998.