Enterprise Architectures – Survey of Practices and Initiatives

Frank Lillehagen and Dag Karlsen
Computas AS, Norway, fli@computas.com and dk@computas.com

ABSTRACT: This paper presents an overview of current practices and development initiatives in the field of Enterprise Architecture, focusing on their underlying concepts and principles. The paper gives a brief historic overview of the field, followed by a description of ongoing initiatives to transform from enterprise architecture methodologies used by consultants and managers to architectures that also guide and support new ways of developing, executing and managing computing solutions.

To conclude the paper presents an outline of existing architecting research approaches, and draws some preliminary conclusions. Based on this survey, some recommendations for future research are discussed.

1 INTRODUCTION

Enterprise architectures are developed for more efficient operations of a business venture. Enterprise architecture *models* are developed to give business managers a better understanding of the things the enterprise owns, operates, and produces so they can make better business decisions. Likewise, industrial and military planners need architecture models so they can understand what they have, what they don't have, and what capabilities they need to have to fill the gaps. This information is used to make better decisions about which capabilities to develop with limited budgets. The features and functions to be provided by hardware and software systems are often described in architecture models in such a way that the proposed systems can be evaluated for how much they contribute to increased effectiveness of business and war-fighting operations.

1.1 *Background*

Enterprise architectures gained popularity when John Zachman developed his framework for dealing with large information systems. [Zachman87] These frameworks originally focused on data processing using mainframe computers, so their emphasis was on the data aspects of a business. Later there was an emphasis on information processing in the business (during the so-called Information Age). In the past decade there has been an increasing interest in "knowledge management" and services, since it has been recognized as a key ingredient in the success of businesses today. However, architecture frameworks still have a focus on the information aspects of the business and little if any treatment of the knowledge aspects.

1.2 *Context*

Enterprise architecture (EA) is a simplified and aggregated representation of the basic structure and organisation of the enterprise. It is a plan which shows the main features and characteristics of the enterprise and its assets that are to be analysed, justified and agreed before the detailed technical design. It is shared and discussed enterprise-wide at a high level of abstraction between all stakeholders.

In the current industrial and economic context, enterprise systems need to be constantly and smoothly re-engineered to respond to changing market demand and technological evolution. Enterprise architecture, considered as the foundation of enterprise systems, has emerged as a 'tool' to help stakeholders to manage system engineering and changes. It is not only an IT issue, but also strategy, business and knowledge ones.

1.3 *Practises*

Enterprise architecture, as an integrating enterprise concept is today developed and delivered as methodology models. For decades, shipbuilding and construction industry have used architecture in the design and construction of their product families. Their "architecture" utilizes standard symbols that can be recognized and understood by all members of their industry for carrying out detailed design, construction and engineering work.

The enterprise engineering community by comparison has never had the advantage of this type of "time tested" structure. Instead, since the beginning, many heterogenous architecture proposals have been developed. They are often overlapping approaches and the underlying concepts are not explicitly defined. Different architecture description languages are not interoperable between them. These languages are either too detailed or lack expressive significance to represent system features. Similarities and differences between architectures can not be perceived by users; and this creates obstacles for its correct understanding in industry and finally its acceptance and use. The lack of a generally agreed terminology and semantically enriched knowledge corpus in this domain is also a bottleneck for its efficient application.

2 ARCHITECTURAL PRACTICES

2.1 *What is an enterprise architecture*

Giving a precise and concise definition on Enterprise Architecture (EA) is challenging. Confronting to the demand of customers, James Martin (2004) lists the following requirements to define EA: (1) must be one sentence; (2) may elaborate from that single sentence to make certain clarifications; (3) must be understandable by executives; (4) must be useful to those who will develop the EA; (5) must be applicable to a government organization; (6) must be put into context of baseline and target EAs; and (7) must provide the executives with the proper expectations of what can be delivered to them and what they can use it for.

Usually, architecture has various meanings depending on its contextual usage (Open Group, 2000): (1) A formal description of a system at component level to guide its implementation; (2) The structure of components, their interrelationships, and the principles and guidelines governing their design and evolution over time; (3) Organizational structure of a system or component.

Generally speaking, EA should be organized in a way that supports reasoning about the structural properties of the system. It defines the components that make up the overall system, and provides a plan from which the system can be developed.

2.1.1 *Enterprise architecture vs. business objectives*

EA does not start with technology, but a strategic framework, the vision, goals, and priority business activities. According to James Martin (2004), an EA is a specific arrangement of business objectives, features and functions. The purpose of a "should-be" (target) EA is to maximize a set of business goals and objectives given a set of constraints, conditions, and challenges. The purpose of an "as-is" architecture (baseline) is to document the current arrangement such that a transition to the desired target state can be determined. Furthermore, the baseline could have been optimized for a prior set of constraints, conditions, and challenges and any changes to these usually demand a change to the current architecture.

2.1.2 *Enterprise architecture as a 'skeleton'*

Independently of business goals or strategies, EA is, first of all, the foundation of enterprise systems. According to ISO 15704 (2000), an architecture is a description of the basic arrangement and connectivity of parts of a system (either a physical or a conceptual object or entity). The software community also considers that architecture is the fundamental organization of a system embodied in its components, their relationships to each other and to the environment and the principles guiding its design and evolution (IEEE 1471, 2000). Specifically, software architecture is 'a set of software components, externally visible properties of those components, and relationships among them' (Florijn, 2002). More generally an architecture must:

- have properties that can be verified with respect to user needs (ex. open or closed architecture, interoperable or not, centralised or decentralised…).
- be simple so that business people can easily understand, check, analyse, discuss as a 'language' shared at corporate level.
- have a style (by comparison with construction where architecture can represent some particular characteristics of a building such as 'gothic', 'romaine'…). EA should be able to characterise enterprise systems (ex. 'fractal', 'holonic', or 'agile'...).

2.2 *Enterprise architecture methodologies*

ISO 15704 (2000) has considered that there are two and only two types of architectures that deal with enterprise integration:

System architectures (sometimes referred to as "type 1" architectures) that deal with the design of a system, e.g. the system part of an overall enterprise integration;

Enterprise-reference projects (sometimes referred to as "type 2" architectures) that deal with the organisation of the development and implementation of a project such as an enterprise integration or other enterprise development programme.

In other words, type 1 architecture represents system or sub-system in terms of its structure and behaviours. The type 2 architecture is actually framework aiming at structuring activities/tasks necessary to design and build a system. For example, Zachman's architecture is type 2 architecture.

Some other works make distinction between *conceptual* and *technical architectures*. The conceptual architecture is derived from business requirements; and are understood and supported by senior management. The technical architecture provides the technical components that enable the business strategies and functions.

Sometimes conceptual architecture is also called *functional* or *business* architecture; and technical architecture, ICT architecture. TOGAF (Open Group, 2000) considers four types of architecture which are subsets of EA: Business architecture, Information Technology architecture, Data/information architecture; and Application (systems) architecture.

Lillehagen et al. (2002) advanced the concept of 'knowledge architecture'. Separation between business, knowledge and ICT architectures should be seen as a design principle (Tinella, 2003).

In software engineering, EA is viewed in a different way. Malan (2002-b) mentioned three types of architectures: (1) *conceptual architecture* (identification of components and allocation of responsibilities to them); (2) *logical architecture* (design of component interactions, connection mechanisms and protocols; interface design and specification…); (3) *execution architecture* (assignment of the runtime component instances to processes; how physical resources are allocated to them). In some other approaches for example (IEEE 1471), enterprise architecture is seen as a complementary architecture to software architecture, to document system-wide organisation and business context in which software operate.

2.3 *The benefits of enterprise architecture*

The concept of architecture is closely related to engineering. There should be significant difference between a system built with architecture and one without. Architecture allows managing complexity and risks due to various factors such as technology, size, interface, context and stakeholders. Design large scale systems (like enterprise) requires high level abstractions. It means we need a mechanism to allow organising enterprise system at sub-system and module levels.

2.3.1 *Architecture vs. stakeholders expectations*

EA at high level of abstraction is a means of communication with and among stakeholders. It allows representing stakeholder's expectations in terms of features of enterprise system rather than documenting detailed requirements on functions, data or resources that will be specified in the later stage.

The role of the architect is to address concerns, show how the concerns and the requirements are going to be addressed, and by showing the trade-offs that are going to be made in reconciling the potentially conflicting concerns of different stakeholders. Without the architecture, it is highly unlikely that all the concerns and requirements will be considered and met (Open Group, 2000).

2.3.2 *Enterprise architecture as a 'blueprint'*

Like in construction, EA allows to create a vision of the future system. This vision is represented as a high level solution structure (Florijn, 2002) that lays down the foundation for design. It is a kind of 'skeleton' focusing on essential features and characteristics of the system. Forces and weakness can be then more easily detected and analysed.

Considering architecture as a high level solution is remarkable. Usually engineering design is divided in preliminary and detailed designs. Preliminary design is also called architecture design. During the preliminary design phase, only solution types are defined. For example, 'prefer matrix organisation than hierarchical one'; 'privilege wireless network than cabled one', etc. Working on high level solution also facilitates the use of deign principles which is another adjacent research subject.

2.4 *Some tentative clarifications*

2.4.1 *Architecture vs. infrastructure*

An infrastructure is not architecture; an infrastructure is a system which may have architecture. At the origin, the concept 'infrastructure' is used in urban engineering referring to some common services of a city. For example, water supplies, electricity network, transport, etc. Similarly, Enterprise infrastructure is a set of services common to all enterprise activities such as 'information exchange, data storage, communication services, local area network, etc.

Ring (2003) also considered that 'An infrastructure provides such common functions such as; a) ensuring the location of modules within whatever coordinate system is pertinent, b) providing access or interconnection, c) providing isolation, and d) ensuring stability throughout a dynamic scenario of traffic, wear, contextual changes and internal innovations'.

Separation between EA and infrastructure is desirable. When EA evolves as business changes, the infrastructure should not be re-engineered every time. On the other hand, when technology evolves, new services can be implemented without the necessity to modify the enterprise architecture.

2.4.2 *Architecture vs. architectural framework*

The term 'architectural framework' is a different concept from system architecture. In (CIO, 1999), an architecture framework is defined as an *organizing mechanism* for managing the development and maintenance of architecture descriptions. TOGAF (Open Group, 2000) also considers architectural framework as a tool to use for developing a broad range of different architectures.

The framework contains: (1) a method for designing a system in terms of a set of building blocks, and for showing how the building blocks fit together; (2) a set of tools and a common vocabulary; (3) a list of recommended standards and compliant products to implement the building blocks.

The term ‘architectural framework’ is also close to another adjacent concept which is ‘*modelling framework*’, for example the CIMOSA modelling framework is type 2 architecture.

2.4.3 *Enterprise architecture vs. enterprise model*

Another adjacent concept to EA is enterprise model (EM). EM describes the EA from various viewpoints in detail to allow specifying and implementing the systems. EA is not executive entity, but an enterprise model can be.

In other words, the focus of EA and enterprise model is different. EA amplifies significant characteristics or features of a system; while enterprise model describes and specifies the system.

Lillehagen (2002) considers that an enterprise model will have an Enterprise Architecture, and Enterprise modelling languages have meta-models that are components of the EA at the knowledge layer.

2.5 *Distinguish architectural decisions*

Another adjacent subject is architectural decisions i.e. decisions to take to arrange architecture in this way rather than that way. Architectural decisions are those that must be made from an overall system perspective. According Bass (1997), those decisions identify the system’s key structural elements, and their relationships, and they define how to achieve significantly architectural requirements. Indeed, enterprise engineering involves many decisions, some are architectural, and most of them are not. The rationale is to show how the decisions are architectural. Malan (2002-a) points out that if you can achieve the requirement by deferring the decision to a lower level, it is not architecturally significant, and the decision is not an architectural one.

3 ARCHITECTURE FRAMEWORKS

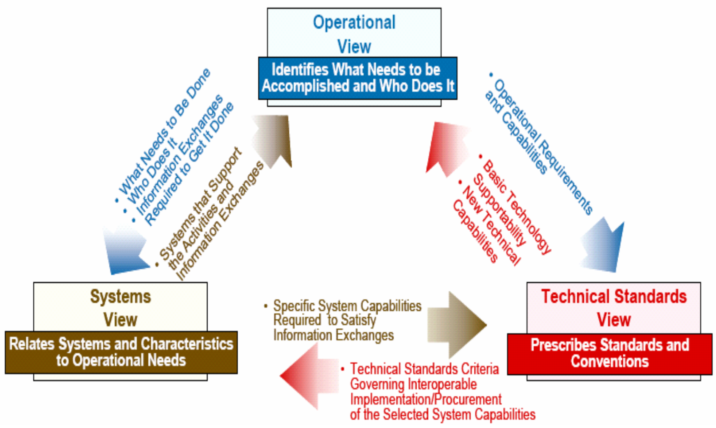
There are many architecture frameworks available to assist the architects in doing their work [Martin 2004]:

(a) Zachman
(b) Gartner Group
(c) TOGAF
(d) SIMA
(e) San Francisco
(f) CORBA
(g) Federal Enterprise
(h) Treasury Enterprise
(i) C4ISR
(j) Archimate

A new framework was in 2003 developed by the US Department of Defense called the DOD Architecture Framework [Dodaf04]. DODAF is an evolutionary upgrade of the C4ISR Architecture Framework both of which prescribe three views of the architecture: Operational, Technical and Systems.

3.1 *The DoDAF Framework*

Architecture products are graphical, textual, and tabular views that are developed in the course of building a particular architecture description. DODAF is composed of three main views as deputed in the figure below.



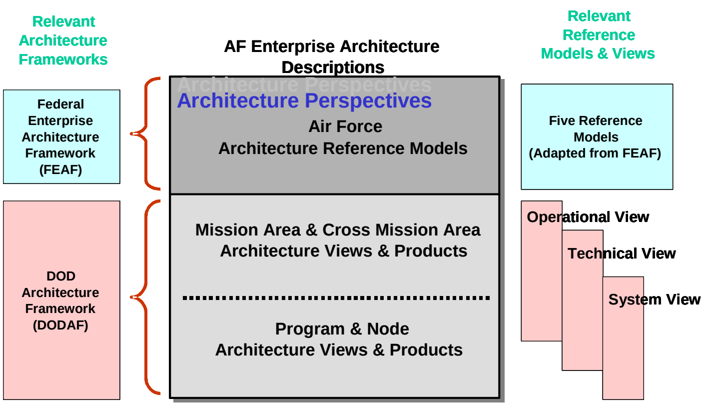
DODAF specifies 26 standard products to be used to develop architecture descriptions with a common approach. The products are shown in the table below.

View Type	Framework Product	Framework Product Name
All Views	AV-1	Overview and Summary Information
	AV-2	Integrated Dictionary
Operational	OV-1	High-Level Operational Concept Graphic
	OV-2	Operational Node Connectivity Description
	OV-3	Operational Information Exchange Matrix
	OV-4	Organizational Relationships Chart
	OV-5	Operational Activity Model
	OV-6a	Operational Rules Model
	OV-6b	Operational State Transition Description
	OV-6c	Operational Event-Trace Description
	OV-7	Logical Data Model
Systems	SV-1	Systems Interface Description
	SV-2	Systems Communications Description
	SV-3	Systems-Systems Matrix
	SV-4	Systems Functionality Description
	SV-5	Operational Activity to Systems Function Traceability Matrix
	SV-6	Systems Data Exchange Matrix
	SV-7	Systems Performance Parameters Matrix
	SV-8	Systems Evolution Description
	SV-9	Systems Technology Forecast
	SV-10a	Systems Rules Model
	SV-10b	Systems State Transition Description
	SV-10c	Systems Event-Trace Description
	SV-11	Physical Schema
Technical	TV-1	Technical Standards Profile
	TV-2	Technical Standards Forecast

The major products in DODAF are related to each other through a set of simple relationships. A change on any product can propagate to other products. It is important to maintain all products in such a way so that they are all consistent with each other.

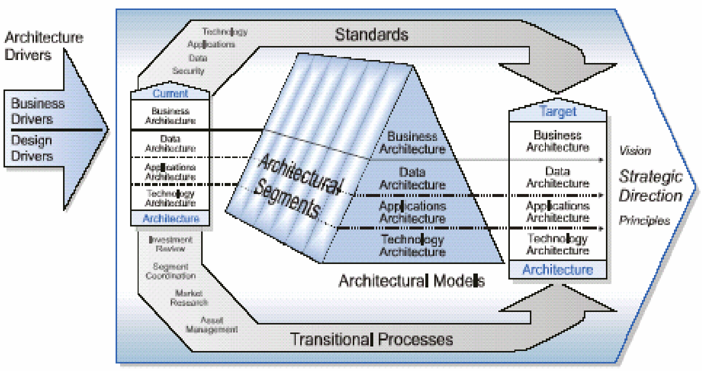
3.2 *The AFEAF Framework*

The US Air Force has recently developed an enterprise architecture framework [AFEAF] to bridge the gap between DODAF and FEAF. The AFEAF has three levels (or "perspectives"): Architecture Reference Models level, Mission Area and Cross-Mission Area level, and Program and Node level. These three perspectives are illustrated below along with how the DODAF and FEAF map into these levels.



The most commonly known framework for enterprise modeling is the Zachman Framework for information systems. [Zachman87, Zachman92, Cook96] The framework is a matrix consisting of six columns and five rows. The columns represent the six interrogatives of **What** (data), **How** (function), **Where** (network), **Who** (people), **When** (time), and **Why** (motivation). The rows represent the five perspectives of the Planner, Owner, Designer, Builder, and Subcontractor.

The Federal Enterprise Architecture Framework provides "an organized structure and a collection of common terms by which Federal segments can integrate their respective architectures into the Federal Enterprise Architecture." [FEAF] The Federal Enterprise Architecture Framework is based on Zachman framework and is shown in the figure below.



Now a relevant question to ask is what extent of "knowledge modeling" is addressed by these frameworks. The result of several analysis is that none of them support active knowledge modeling, that is they are all collections of static information views. They are offline methodologies with no direct connection to systems execution or business or field operations.

4 ARCHITECTURAL APPROACHES

4.1 *Architectures for enterprise integration)*

The architecture frameworks are concerned with concepts and principles for integrated enterprise modelling and engineering. They do not deal with the representation of how an enterprise is structured or operated. In other words, they do not represent processes, data, organisation etc. Rather, they define the key concepts that are necessary to describe enterprise systems so that models built are consistent and easily integrated across stakeholders.

4.1.1 *Early architectural initiatives*

During 80's, research were carried out in Europe and U.S.A. to develop enterprise architectures. Among them the most known: the Purdue Enterprise-Reference Architecture (PERA) (Williams 1991), the GIM architecture (Chen and Doumeingts, 1996), the Computer Integrated Manufacturing Open System Architecture (CIMOSA) (Amice 1993) etc. These architectures are mainly developed along the system life cycle to show what must be done to model, design and implement an integrated enterprise system.

The Zachman architecture is another example of these initiatives. It structures various enterprise modelling and engineering concepts according to 'the perspectives of various stakeholders involved in the enterprise engineering effort. This is because different stakeholders use different levels of abstraction to consider an enterprise and expect different deliverables.

4.1.2 *IFAC/IFIP Task force to develop GERAM*

The IFAC/IFIP Task Force on architecture for enterprise integration has studied some existing architectures to propose harmonization. The result is Generalised Enterprise-Reference Architecture and Methodology (GERAM) (Bernus, 1997). GERAM identifies, in its most important component called GERA (Generalised Enterprise Reference Architecture), the basic concepts to be used in enterprise engineering and integration.

4.1.3 *Standardization works*

Two institutional standardization bodies active to develop type 2 architecture standards are: CEN TC310/WG1 and ISO TC184 SC5/WG1. Main outputs achieved are: ISO 15704 (1998) - Requirements for Enterprise Reference Architecture and Methodologies, and ISO/EN I9439 (2003) - Enterprise Integration – Framework for Enterprise Modelling. However, these works are not well known in industry and not widely used.

4.1.4 *New Initiatives*

Another significant framework is TOGAF. It contains two main parts: The Architecture Development Method (ADM) and Foundation Architecture with generic functions or services on which specific architectures and building blocks can be built. The Architecture Development Method is a recommended approach to building TOGAF compliant architectures.

All the frameworks and approaches mentioned

4.2 *Architectures of enterprise systems*

There are few works done at the conceptual level concerning system architectures, except some partial approaches dealing with for example, manufacturing data (ISO 15531, MANDATE), enterprise-control system interfaces (IEC 62264), etc. These approaches are developed in a very detailed level so that they are more reference models rather than enterprise system architectures. More recently the European Technical Specification (CEN TS 14818: Decisional Reference Model) was approved. It is based on the GRAI approach and shows a basic decision-making structure defined at a fairly high level abstraction.

Most of bottom-up system approaches are concerned with ICT architectures or infrastructures. For example, ENV 13550 - Enterprise Model Execution and Integration Services (EMEIS) is a European experimental standard. It focuses on services to allow execution of an enterprise model to control industrial processes.

Other standards concerning IT architectures are: ISO 13281.2 -Manufacturing Automation Programming Environment (MAPLE), Open Management Architecture (OMA) known as CORBA developed by Open Management Group.

The functional architecture of MAPLE focuses on manufacturing application interconnection and interoperation, it is not an enterprise-wide architecture. CORBA architecture also supports application integration focusing on issues affecting distributed object-oriented systems.

Some research work towards intelligent infrastructure has been reported in (Lillehagen, 2002). An Intelligent Infrastructure is a layered representation of knowledge in various forms and formats. It can cover not only the ICT layer, but also the knowledge and business layers.

5 ARCHITECTING PRINCIPLES

Architecting principles are rules to use when elaborating enterprise architectures. Architecting principles may be generic i.e. apply to all enterprises (reflecting some best practices), or specific (reflecting a level of consensus among the various elements of a particular enterprise, and form the basis for making future decisions). Sometimes too many principles can reduce the flexibility. Many organizations prefer to define only high level principles, and to limit the number to between 10 and 20 (Open Group, 2002).

5.1 *General principles*

General enterprise architecture principles can be found in the literature such as for example (UNCC, 2003): (a) Business processes drive technical infrastructure, (b) Primary purpose of architecture is to facilitate rapid change, (c) EA must emphasize reusable component building blocks, (d) Architecture must be enterprise-wide, (e) EA must optimize the enterprise system as a whole, etc.

Open Group has proposed similar kind of principles ranging from business to data, application and technical principles (Open Group, 2002). Once again these principles are at high level of abstraction. They are no directly related to architecture features and properties. Moreover, they don't provide examples showing how to implement the principles.

Another approach can be found in the Government of Canada's Federated Architecture (2001) where some principles were proposed, for example: (1) *Reduce integration complexity* to re-engineer application systems to be "highly modular" and "loosely coupled" to be able to reuse components; (2) *Adopt holistic approach* with a (whole of enterprise) approach; (3) *Business event-driven* systems; (4) *Plan for growth* and construct for growth and expansion of services (known requirements) across enterprise; (5) *Robustness*, responsive, and reliable with appropriate redundancy to protect against system failure, etc.

More generally speaking, when developing EA, the principle of fitness-for-purpose should be followed. It means that the architecture should be developed only to the point at which it is fit for purpose.

5.2 *Technical principles*

Cockburn (2003) have proposed some architecting design principles, for example: (1) *Create an interface around predicted points of variation* (because things change, and we must protect the system integrity across changes); (2) *Separate subsystems by staff skill requirements*; (3) *Make one owner for each deliverable* (People get confused if ownership is unclear); (4) *The program is totally program driven, with the user interface just one driving program* (because user interface requirements change a lot); (5) *Provide a single point of call to volatile inter-team interfaces* (Protect developers against rework due to an interface change), etc.

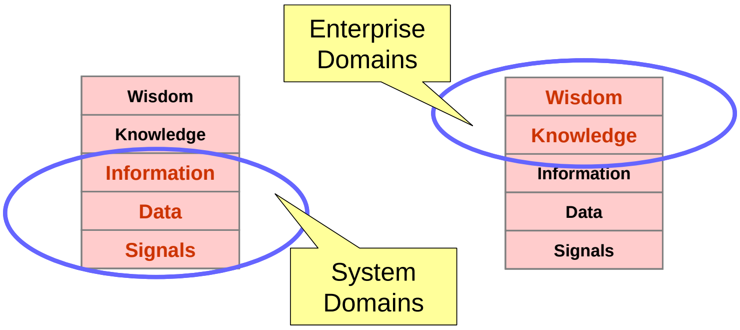
Malan (2002-b) also suggested three principles to develop a 'Minimalist Architecture': (1) if a decision could be made at a more narrow scope, defer it to the person or team who is responsible for that scope; (2) only address architectural decisions at high-priority architecturally significant requirements; (3) as decisions are added to the architecture, they should be evaluated from the point of view of their impact on the overall ability of the organization to adopt the architecture.

In TOGAF (2000) some principles underlying the design and successful use of specific architectures were proposed, for example: (1) An architecture need only specify those services that it requires; (2) Elements of an architecture may specify one, more than one, or only part of a service; (3) Elements of an architecture should be defined in terms of standards relevant to the services they specify; (4) Elements of an architecture should be reused from all the categories of the Architecture Continuum and should support reuse of solution elements of the Solution Continuum; (5) Elements of the solution (or implementation) should be reused from all the categories of the Solutions Continuum; (5) An architecture must be followed, or it is useless: formal IT Governance practices are therefore recommended.

6 ACTIVE KNOWLEDGE MODELING

An active knowledge modeling approach will in the future be part of most architecture frameworks, approaches and solutions.

Existing frameworks and approaches are mostly describing abstracted views of the systems domains, but many initiatives will be launched in 2005 to develop approaches and platforms for business and knowledge architectures and enterprise domains.



The following steps summarize some proposed approaches towards developing active knowledge modeling support for architectures design and operational architecting:

1) Provide an integrated visual knowledge modeling and execution platform
2) Develop services for active knowledge modeling, supporting design and re-iteration
3) Develop services for model-driven solutions and work execution
4) Develop knowledge management services
5) Re-engineer and integrate present framework domains and views
6) Mode-generate easy to use workplaces for all stakeholders to engage in architecting, operating and managing their knowledge

The major contribution of this research and development is the integration of a knowledge modeling techniques and services, and the introduction of a common platform for use in development of enterprise architectures and solutions and work execution.

7 RECOMMENDATIONS FOR RESEARCH

Research is needed on active knowledge modeling as part of an overall architecture development process. The following steps summarize a proposed approach for introducing active knowledge modeling and integrated modeling and execution platforms.

1) Understand the nature, context and benefits of active knowledge modeling
2) Define a methodology and language for active knowledge modeling
3) Develop a knowledge modeling technique, an integrating platform and services
4) Apply approach to the existing Architecture Framework models
5) Evaluate suitability and benefits of approach and solution.

These steps are being considered by partners and stakeholders that have experiences and competence in both TOGAF and DODAF. Concerning enterprise systems architectures, more development of some reference architectures at higher abstraction levels is also needed. This would help re-use of some mature solution types to save time and cost of enterprise engineering projects.

It seems interesting to develop simple architecture representation language that allows to better exhibit key characteristics of enterprise systems at high abstraction level. Current enterprise modeling languages (IDEF for example) seem not well adapted to represent architectures features. Differentiation and clarification between enterprise architecting and enterprise modeling are necessary using some concrete examples. The two approaches and application areas are closely related.

Moreover differentiations between various architectures in terms of properties and features also need to be explicitly identified so that comparison and choice can be more easily made at a high level of abstraction and early stage of design. The frameworks and their views must become active or living knowledge related to work throughout life-cycles.

It is also necessary to continue the effort of harmonization. At least, a standard terminology is needed. This requires establishing collaboration between enterprise modeling community developing business oriented architectures, and software engineering people concerning the IT oriented ones.

Architecture design principles were not developed to a satisfactory level that brings significant improvement to enterprise architecting. Further research is also needed in this area. Developing architecting principles can be bottom-up based on best practices, and/or top-down by studying some theoretical paradigms.

8 CONCLUSIONS

This paper has tentatively reviewed basic concepts and principles of enterprise architectures. A survey on some main architectural frameworks and approaches is presented. To day, the architecture concept is not sufficiently exploited as a high level solution structure defining solutions types. One of the reasons is the lack of proper architecture representation formalisms providing characteristic features and properties of enterprise knowledge dimensions and domains.

ACKNOWLEDGEMENT

This paper is partly funded by the E.C. through the Interop NoE. It does not represent the view of the EU nor the Interop NoE project.
The figures are taken from papers by James Martin, US Air Force. The authors are
The authors are responsible for the paper’s content.

9 REFERENCES

AMICE (1993), *CIMOSA - Open System Architecture for CIM*, 2nd edition. Springer-Verlag, Berlin.

Bass, L., Clements, R. and Kazman, R. *Software Architecture, in Practice*, Addison-Wesley, 1997.

Bernus, P. and Nemes, L. (1997), The Contribution of GERAM to Consensus in the Area of Enterprise Integration, in: Kosanke K. and Nell J.G. (Eds.) Enterprise Engineering and Integration: Building International Consensus, Springer-Verlag, Berlin, pp. 175-189.

Chen, D., Doumeingts, G. (1996), The GRAI-GIM reference model, architecture and methodology, in: Bernus P. et al. (Eds.) Architectures for Enterprise Integration, Chapman & Hall, London.

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

Cockburn, A. (2002), On the Interaction of Social Issues and Software Architecture, http://members.aol.com/acockburn, Humans and Technology (arc@acm.org).

ENV 13550 (1995), Enterprise Model Execution and Integration Services (EMEIS), CEN, Brussels.

Florijn, G. (2002), Software Architecture - Introduction & Overview, (slides), SERC, 2002.

Government of Canada (2001), www.cio-dpi.gc.ca/fap-paf/documents/iteration/iterationtb_e.asp

IEEE 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems.

ISO 15704:2000(E), Industrial automation systems - Requirements for enterprise-reference architectures and methodologies.

ISO/EN I9439 (2003) - Enterprise Integration – Framework for Enterprise Modelling.

Lillehagen, F. & Krogstie, J. 2002, Active Knowledge Models and Enterprise Knowledge Management, *in Enterprise Inter- and intra-organizational integration*, Kluwer Academic Publishers, Kosanke K. et al (ed.)

Lillehagen, Frank (2003a) and Helge Solheim, “The Foundations of AKM Technology,” Concurrent Engineering Conference 2003, Madeira, July 2003.

Lillehagen, Frank (2003b) and J. Kostie, S. Tinella, H.G. Solheim, D. Karlsen, “From Enterprise Modeling to Enterprise Visual Scenes,” Concurrent Engineering Conference 2003.

Malan, R. and Bredemeyer, D. (2002-a), Less is More with Minimalist Architecture, *IT Professional IEEE Computer Society*, September/October 2002.

Malan, R and Bredemeyer, D. (2002-b), Software Architecture: Central Concerns, Key Decisions, *Software Architecture*, 2002 BREDEMEYER Consulting.

Martin, J. (2004), Enterprise architecture - Definition, internal communication note, 2004.

Martin J. (2005). Proposal for Doctoral Dissertation, private communications with the author.

Open Group (2000), TOGAF: The Open Group Architecture Framework, Document No. 1910, Version 6, December.

Open Group (2002), www.opengroup.org, Architectural Principles, 2003.

Ring, J. (2003), E-Business Infrastructure Capability Assessment, 2003.

Tinella S. et al (2003), Model Driven Operational Solution: The User Environment Portal Server, *CE 2003*, Madeira.

UNCC (2003), Architecture, www.uncc.edu.

Williams, T.J. (1991), The Purdue Enterprise Reference Architecture, Instrument Society of America, Research Triangle Park, North Carolina.