Architecture and System Design Issues of Contemporary Web-based Information Systems

B. Molnár[1], Á. Tarcsi[2]

[1]Corvinus University of Budapest, 1093, Budapest Fővám tér 13-15., Sóház II/225, Hungary
[2]Eötvös Loránd University, Faculty of Informatics, Department of Information Systems, 1117, Budapest, Pázmány Péter sétány 1/c. Hungary,

The rapid changes of information technology led to the proliferation of Web-based Information Systems (WIS). There has been already some research on the general model for analysis and design of WIS. This paper attempts to define a comprehensive framework for integrating the various viewpoints of model for WIS. Several components that compose of a WIS as analyzing and designing element are investigated and described as modeling element and enabling tool for creating consistent and integrated system. The major modeling components are: the Web site, semi structured document, business processes, element of knowledge management, the enterprise and information architecture, and autonomous software components providing functional services. These elements are integrated into a unified framework exploiting partly the object-oriented paradigm and partly other approaches for information system modeling. The outlined WIS model incorporates the most recent approaches for various paradigms of traditional information systems analysis and the most modern different approaches of WIS. The research result would be useful both theoretically and practically as it could present a comprehensive model for WIS assisting our understanding of properties of WIS and supporting, at the same time, a framework for being able to formulate a practical development method for achievement of WIS by organizations making the most of Web.

Index Terms— **Object oriented modeling, Object oriented methods, Software architecture, Web services.**

I. INTRODUCTION

THE PAST DECADES has presented a dramatic change of the use of information technology within organizations and enterprises. To control the inherent complexity of structure of large-scale information systems and information resources, various IS architectures have been elaborated to describe and to keep in hand the several layers, tiers and components of architecture.

Architecture provides a kind of synthesis through providing views and viewpoints that can be realized by various models proposed in different methodologies for architecture and information systems analysis and design.

A *web information system* (*WIS*) is database-intensive and/or transaction intensive information system that is implemented and made accessible over the web by user access via web browsers. In web information system, data is obtainable through web interfaces that are manifested in semi-structured document format including a navigation structure among the documents and pages organized into some kind of networks or hierarchy; moreover there are links to sites outside the system as well. Furthermore, there should also be services for retrieving and modifying data out of the system or from the underlying database.

Service-Oriented Architectures (*SOA*) have appeared after the WIS being in existence for a while; they achieved large success within the Web communities, especially at WIS. Rather than pure technology improvements SOA intends to increase manageability and sustainability of information systems and to better align business requirements and technology implementations.

Therefore it is reasonable to link SOA to the concept of *enterprise architecture* (*EA*). As the enterprise architecture provides a global framework to deal with the inherent problem of WIS modeling, design and operation.

In addition, the change processes occurring during the life of a WIS dedicated to an organization are typically concurrent and interact with one another. To handle the complexity of changes and change management some concepts are required namely the concept of the *life history,* the *life cycle* of systems, and the *life history of entities* or *classes (of objects)*. We should use these concepts to distinguish between two behavioral aspects of WIS. The terms of life history of systems means "the actual sequence of steps a system has gone or will go through during its lifetime", and life cycle seen as ''the finite set of generic phases and steps a system may go through over its entire life history'' [1]. Thus, life history and life cycle are in fact orthogonal, as shown in a typical life history example is provided in [2]. A WIS fitting into the Enterprise Architecture of an organization has a static perspective representing the structural aspects at a specified point in time and a dynamic perspective capturing the various stages of architecture development that happens to a WIS as an application system [2]. These two perspectives may reflect the changes that take place at component level of architecture of a WIS – both statically and dynamically - , however there is a need to represent the time dimension at components of development, implementation and operation level, i.e. the *life history of components* or *entities*.

Although the development of the logic of a Web Application is similar to the conventional software development the process of implementing the web-based

Manuscript received April 18, 2011. The Project is supported by the European Union and co-financed by the European Social Fund (grant agreement no. TAMOP 4.2.1./B-09/1/KMR-2010-0003).
Bálint Molnár is with Corvinus University of Budapest, 1093, Budapest Fővám tér 13-15., Sóház II/225, Hungary (corresponding author e-mail: (molnar@infomatika.uni-corvinus.hu).

information systems behind these applications has, however, shown the deficiencies of the existing methodologies for modeling, analysis and design and the need for the specification, design and implementation of new mechanisms.

The WIS contains traditional information systems element and semi-structured XML documents which behaves actively beside other important parts of the system. The XML documents make use of active hyperlinks and other features of Web and Web services.

This paper wants to elaborate a modeling framework for WIS taking into account of the development of past decades considering the various architectural and technological component and solution. We discuss the proposed model and show how the earlier created methodologies and methods for analysis, design, implementation and management can be used to construct web-based applications in a framework to combine aspects of data, behavior, state and presentation and exploit the various perspectives and aspects of approaches to make consistent information systems.

An attempt is made in this research to create a (1) firstly a comprehensive model including the relevant aspects of WIS, (2) based on the model a methodology and related methods to support the design process.

II. Related Works

There were already attempts to put into a unified framework the modeling of WIS at a given point in time taking into account of the existing technology and approaches. A general model for WIS using an object-oriented approach is described in [3]. The use of semi-structured, active XML documents and a disciplined design approach are discussed in [5] to construct web-based applications. Another paper [6] presents a design methodology for a systematic design process to organize and maintain large amounts of data in a Web site, in the form of a hypermedia design methodologies. For large-scale WIS design [10] contains a method.

Beside the different analysis and design model for WISs, the concept of architectural approach is that can be used in model creation, the various IS architectures have been used to assist in understanding the relationship between the various perspectives, aspects, components and single models [7], [1], [2]. Zachman architectures developed for information systems. TOGAF is developed by Open Group on Architecture Framework. It contains two main parts: The Architecture Development Method (ADM) and the Foundation Architecture with generic functions/services on which specific architectures and building blocks can be built [8].

The service orientation is emerging at multiple organizational levels in business, and it leverages technology. The service-oriented architecture (SOA) can be considered as a technical architecture, a business modeling concept, a piece of infrastructure, an integration source, and thereby an architecture for contemporary WIS [9].

While each of the above approaches overlaps with some of the features of our work, the contribution we make is to address at problems that concludes the dynamical behavioral of WIS-s in several aspects and how to handle at modeling level the issues of consistency, integration, confidentiality, accuracy and timelines.

III. Information System Architecture

The *Information System Architecture* represents the structure of certain components of information processing systems, their relationships among components, those technological principles and directives of organization of which the main purpose is to support business.

The description of how a system was internally built depicted primarily the technological components, especially the software building blocks. The *Zachman Framework* ([7]) can be considered as the first important phenomenon that pinpointed to the fact that software architectures were not enough. While software architectures represent internal system details (using, for example, E-R and DFD diagrams) *Information System Architecture* focus on the high-level business and information system processes.

We perceive *Enterprise IT Architecture* as the suite of strategic and architectural disciplines that includes the Information, Business System, and Technical Architectures.

An *Enterprise Architecture* can be divided into several levels:

Business (systems) architecture - Defines the structure and content (information and function) of all business systems in the organization.

Information (or Data) Architecture – represents main data types that support business; furthermore the structure (including interdependencies and relationships) of information required and in use by the organization;

Application Architecture – defines applications needed for data management and business support; the collection of relevant decisions about the organization (structure) of a software system, and the architectural style that guides this organization.

Technical Architecture – represents the main technologies used in application implementation and the infrastructures that provide an environment for information system deployment. Technical architecture describes and maintains the integrity of the hardware, software, and infrastructure environment required to support the *Business Systems Architecture* and *Information Systems Architecture.*

A. Control architecture: The product of dynamic modeling

Control architecture provides a temporal view on the dynamics of data, application, and technology. The temporal dimension and behavior of data and/or information reflected in the form of entity life history, life cycle, state chart or state transition diagram.

However, there are no established and widely accepted formalism and approaches at developmental control, operational control, and maintenance control to keep in hand the ever changing environment and provide a descriptive method to support handling the related issues.

Developmental control handles all the changes occurring in the process of new application development over time. Also called version control or *software configuration management*, developmental control records various aspects (who, what, and when) of each change made.

Operational control concerns the performance and integrity of current data, applications, and technology configuration.

In WIS, the changes may happen at *both document* (XML / HTML) level and *data* level, sometimes parallel; both changes indicates alteration on the application (logic) level as well.

This is when *maintenance control* becomes a critical issue. How can we manage these changes without severely disrupting the existing operations of the business? What will be the impact of a certain change to specific parts of the organization? Who are in charge of what? Some of these are included in the "impact analysis" portions of modern data dictionary systems. The business processes within information function of organization should provide adequate answers within the ITIL (Information Technology Infrastructure Library, [17]).

TABLE 1 HERE

B. *Service-Oriented Architectures (SOA) and the WIS*

Service-oriented computing (SOC) is considered as a new information technology paradigm after the object-oriented paradigm. It utilizes services as fundamental, reusable elements for developing applications / solutions. One of the basic components of SOA approach is the concept of services that maps the business services to information services in the form of *Web-services*.

The SOC paradigm and Web-services together provide an appropriate approach for building Web based information systems (WIS) that a definite feature of the collaboration i.e. the comprising element work together in an interactive manner within a dynamically changing environment.

The WIS-s making use of SOC have - as an advantage - the direct connection to the organization business processes, workflows, activities and tasks. The *Business Process Modeling* (BPM) realizes this direct mapping between the organizational level services and the information system level services. The Business Process Execution Language for Web Services (BPEL) is an example for an executable language that implements the results of BPM.

As the size of Web applications grows, it becomes clear that better models, methods, methodologies and tools are required to deal with their increasing complexity.

A model and a design methodology of WIS promotes the idea that the large and complex Information Systems made available by the Web technology should be modeled and designed as large scale Information Systems embedded into business and organizational environment. The model and design methodology should take into account both the socio-technological environment and the Web and / or software engineering approaches.

WIS analysis, modeling and design methodology should include

1. aspects of computing on the Internet,
2. distributed transaction processing,
3. knowledge management,
4. hypermedia - the Web site,
5. intranets,
6. and extranets
7. XML / HTML documents either as Web pages or information resources,
8. database management systems and data warehouses,
9. document management systems,
10. Enterprise Architecture, Service Oriented Architecture, Web services.
    a. Mediator;
    b. wrapper architecture for Web applications;
    c. Web servies - software agents.
11. Business processes analysis and management the Web site - online business processing.
12. Information Security: integrity, confidentiality, accuracy, availability, timeliness - information infrastructure.
13. Performance and scalability issues.

The core of WIS is an information system in the traditional sense. Models, analysis and design approaches of conventional information system are evolved trough decades, the most important modeling approaches applied nowadays include:

1. Data-flow diagram method ([20],[21],[22]);
2. entity-relationship method ([23]);
3. Relational analysis;
4. Functional analysis, hierarchy and decomposition;
5. Business and information event analysis and mapping to state transition of the conceptual information and/or data model.

The WIS's goals can be formulated as to support networked organizations in the integration of specialized Web sites into a common set of tasks for them and providing organizational computing network properties. The overlapping of WIS and conventional business IS appears in the form of data processing systems, data base management, report systems, and decision support systems.

The difference between a set of Web pages, typical Web Applications and a WIS can be described as follows: WIS supports business process (Business Process Modeling, BPM) and is usually tightly integrated with other IS. WIS can also be viewed as database applications for structured as well as semi-structured data (XML / HTML).

There is a comprehensive meta-model for WIS ([3]) that overarching all views, aspects and perspectives from semi-structured documents to knowledge management. There are models in object-oriented style for all element of Wang's meta-model and the components of meta-model can perceived as artifacts of Zachman architecture and put into the proper cells (Table 2).

The semi-structured documents embodying hypermedia or hypertext document plays a crucial role in the most modern IS as WIS ([24],[25], [26]). Historically documents have been used in displaying, interchanging, and retaining information. The so-called form documents have been used in traditional IS as well, originally for presenting user interfaces, displaying the underlying data structures, and demonstrating the related transactions; later on, the form documents have become the typical layout for the WIS user interfaces using the HTML standard.

Within the context of an organization, the activities of creation, exchange and modification of documents comprise a series of tasks within business processes that all together makes workflows. Each relevant workflow can be represented as directed acyclic graph having tasks or information processes in its nodes. The document flow within business can be mapped to the information flow of a single workflow that is implemented by information technology; previously the base of realization was the traditional IS, nowadays the business processes are carried out by WIS. This phenomenon can be summarized as evolution of the information processing paradigm, shifting from the process-oriented, data intensive application typically represented and designed in object-oriented style to *document-oriented* computing. The development of information technology has led to the active semi-structured documents. The documents – e.g. the form documents intended to collect data –, contain procedures for information processing as it is required at a given time at a certain position within the workflow, thereby the documents demonstrate active behavior.

FIG. 1 HERE

C. *Architectural viewpoint*

An overall architecture approach assists in creating and conceptualizing a comprehensive model for WIS. As WIS not only a variant of information systems but it is a new technological approach for realizing information system.

The most modern architectural approach is SOA that slowly becomes the typical basic architecture for a large number of WIS. The mapping the elements, models are outlined in Table 1, Table 2.

D. *WIS Functional Viewpoint*

The business processes carried out by WIS can be described by object-oriented analysis and design methods. The business processes realized by WIS demonstrate the functional side or functional services of IS.

A business process can be modeled by using three fundamental types of object classes:

– Conceptual mapping of entities or *concepts*, - objects represent the structured data in WIS;
– *event*, - represent events of routine operations (e.g., order processing) , decision activities;
– *document* - information entities that enter the system (e.g., electronic order applications), or that are produced by the system (e.g., online business reports and client side scripting XML/HTML documents), or stored the system as semi-structured data (XML documents).

Beside the above mentioned parts the WIS generally contains knowledge management components as well, namely, organizational learning, mapping some cognitive activities onto a structures representing cognitive properties ("cognizant" element [3]): production rules, semantic networks, cognitive maps, or other forms of knowledge and knowledge representation.

All of the models covering each single part of WIS use the object-oriented paradigm making comparable, verifiable, through cross-checking pair of models. The object-oriented framework provides an explicit logical schema of WIS, and can be tailored and actualized based on the requirements of a specific WIS.

E. *Integration within the tier of control architecture*

We have now at least five separate dimension or aspect of IS (Fig. 1). An optional synthesis of the five aspects can be formalized into a dynamic depiction as it follows:

A business process starts with Web page descriptions, presentation or design. Each of the operators of the Web page initiates messages to other types of objects. The four possible outcomes in response to an initial message:

a) The *message* from the Web site object invokes another Web site object.
b) The message triggers a *knowledge object,* which might be an element for representation of knowledge.
c) The message evokes a *software agent* or *Web service* (e.g., a search engine), which might in turn trigger a *business process.*
d) The message passes through a network or Enterprise Bus Service, and then starts a *Service* realizing a business process, usually, by triggering an *event* of the business process.
e) After a business process starts, its *entity* (conceptual mapping of physical real-world objects), *document*, and *event* objects are triggered subsequently.
f) Each of the objects of the business process activates the corresponding *architecture* elements (network service, other service component, logical, physical application or information component, etc).
g) The objects of the architecture elements send messages to other information architecture objects to verify the components of the information infrastructure whether there are or were any breach of consistency or security.

TABLE 2 HERE

F. *WIS Semi-Structured Document Viewpoint*

The Web is playing a critical role as a source of information. Most of Web-based applications are document centric. Most of business processes in an organization are carried out through workflows making use of documents as information source and transmitter. Moreover, the Internet technology accelerates business processes to be implemented by Web-based applications. From the viewpoint of human-computer interaction, Web-based document actualize the structured and semi-structured data through the computer-user interface of the WIS, and specify the dialogue between users (clients) and the WIS. A central part of WIS is the Web pages and Web semi-structured data in the form of XML/HTML documents for the users on the network (Intranet, Extranet, Internet). Presentations of information to the users and requests for input from the users are the two facets of Web sites from the viewpoint of the WIS. With the documents implemented in extensible markup language (XML), the relationship and information exchange between the Web pages and the associated business processes and knowledge representation is realized.

G. *WIS Dynamic Behavior Viewpoint*

The intrinsic nature of WIS is the dynamic behavior in every component, the user interface and its semi-structured document and the underlying IS, but not only in respect the data content however the data an processing structure as well.

Behavior: A behavior means a method of processing a document implied in a business document. Behaviors include (1) general computation logics, (2) rules of defining and controlling business processes such as business rules, (3) data integrity constraints necessary for validation of data, and (4) workflow execution.

Behavioral correctness: The correctness of behavioral composition is concerned with the security, safety, faithfulness and fairness of behavioral properties. The composition of two components should behave properly. The composition may mean the creating or modifying a semi-structured document then storing , or linking to the data structure and content of underlying IS. The composition may mean the orchestration and / or choreography of service component. The composition could mean storing of the semi-structured document in native format, independently from the data content and structure of underlying IS and then later-on could be reconciled and integrated to eliminate the inconsistencies between the documents and WIS. To resolve the inconsistencies a name mapping associates the classes and objects declared in a component with the classes and objects declared in the composition of this component and other components.

H. *Consistency and Reconciliation of Model Element*

Having seen previously the heterogeneous approaches and models that reflect important aspects of WIS, we encounter the question what general modeling framework can be employed to describe a comprehensive methodology for modeling and design that provides the option for further development in the sense of refinement and towards a semi-formal and later-on a formal description.

To put the disparate and different models into a unified framework, the first attempt is to map the WIS and its possible models into the Zachman architecture that contains the important aspects and views of an IS within an organization context. To make the various models comparable, verifiable in the sense of correctness, the object-oriented modeling approach can be used for each modeling artifact as all the views of Zachman can be modeled by object-oriented paradigm from the classical IS and database technology to the document-oriented WIS.

Axiomatic Design Theory ([27], [28]) is a systematic methodology that assists designers to structure design problems. Originally, AD identifies four concepts, namely: domains, hierarchy of design artifacts, zigzagging between domains and element of hierarchy, two axioms. *Independence Axiom*: Maintain the independence of the functional requirements; *Information Axiom*: Minimize the information content of the design. The axiomatic design can be applied to modeling and design of systems that consist of components that present comprehensible complexity, the components are independent from each other and the requirements can be unambiguously distinguished from each other. There are four domains: the customer domain, the functional domain, the physical domain, and the process domain.

Instead of the originally defined domains of axiomatic design, we will use the perspectives of Zachman architecture as "domains" as all domains of axiomatic design theory is contained within the set of Zachman's perspectives and represents the life cycle of system development and operation.

The artifacts that describe *what*, *how*, *where* and *why* the features and systems of features operate in the logical and physical domains can be defined in an explicit, well thought-out manner using of object-oriented paradigm. The framework matrix consists of a vertical axis that provides multiple *perspectives* of the overall architecture and a horizontal axis, which provides a *classification* of the various *artifacts* within the architecture. The applicable perspectives and classifications within the framework are illustrated in Table 2.

The advantage of object-oriented approach is that it operates as a *lingua franca* at syntactical level of modeling and there are modeling approach in object-oriented paradigm for each artifacts of Zachman architecture.

As to axiomatic design theory, the elements within each domain are Customer Needs (CNs), Functional Requirements (FRs), Design Parameters (DPs), and Process Variables (PVs).

The axiomatic design methods provide a powerful design documentation tool that can be easily understood by people uninvolved in the process. Not only is it clear which DPs satisfy which FRs, but it is easily understood how the system has been decomposed, and the level of coupling within the design.

The mapping process between the domains can be expressed mathematically in terms of the characteristic vectors that define the design goals and design solutions. At a given level of design hierarchy, the set functional requirements that define the specific design goals constitutes a vector {FRs} in the functional *aspects*. Similarly, the set of design parameters in one of the perspectives for the FRs also constitutes a vector {DPs}. The relationship between these two vectors can be written as:

$$\{\mathrm{FRs}\} = \mathrm{A}\{\mathrm{DPs}\} \qquad (1)$$

The matrix *A* type represents the actual "method", the transformation and mapping between the functional requirement and design decision (**1**), the FRs, the functional requirements appears in the higher level *rows*, i.e. the *perspectives* of Zachman architecture. The verification and validation for properties of *correctness*, *faithfulness*, *consistency* and *integrity* among the design decision is represented by matrix *B* type (**4**). Matrix *B* signifies the transformation, the reconciliation and resolution of contradicting constraints and parameters between the relevant pairs of artifacts or models situated in the *columns*, i.e. *aspects* of Zachman architecture.

$$[A]=\begin{bmatrix} a_{12} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad (2)$$

The User Requirements, the *Customer Needs* (*CNs*) transformed into *Functional Requirements* (*FR*s). Each single *FR* can be represented by objects within the aprrropriate model prescribed by the architecture and joining method in Table 2. The DPs can denote input, output data, constraints, event parameters. Different levels of FRs are represented as objects with indices reflecting the design hierarchy. The perspectives of Zachman's architecture describe the typical life cycle and hierarchy of model and design artifacts. Therefore, the FRs at a higher level after transformation by DPs appear as FRs for lower level design artifacts.

$$\{DPs\}=B\{DPs\} \quad (3)$$

$$\{DPs\}=C\{PVs\} \quad (4)$$

As all functional requirements are not at the same level. There are hierarchies in FR, DP and PV as parameter domains of axiomatic design, which are created through decomposition. However, they cannot be decomposed by remaining in one domain of axiomatic design. One must zigzag between the domains of axiomatic design and between the artefacts of aspects and perspectives of Zachman's architecture to be able to decompose the FRs, DPs and PVs. Through this zigzagging we create hierarchies for FRs, DPs and PVs in each parameter domain of axiomatic design. Through this decomposition process, we establish a set of hierarchies of FR, DP and PV. The objects created by the modelling and design process embody the requirements, parameters and variables at various level and at different aspects.

TABLE 3 HERE

The object in a model exhibits some well-defined behavior that is modeled by some method describing the state changes of object. Whatever be the object either entity in the database or element of semi-structured document or cognizant component of WIS, the behavior plays a crucial role during the whole life cycle of WIS either modeling and design time or operation time. Behavior is a special case of *FR* . The relationship between objects and behavior can be perceived to the decomposition of *FRs* in the *FR* hierarchy of axiomatic design. Object can be regarded as the *parent* FR relative to object presenting *behavior* which is the *child FR* . That is, the higher level FR (contextual and conceptual level of perspectives) in the form of objects within the appropriate model show rather static objects and the children FRs of the object FR demonstrates rather dynamic objects in the form of "*behavior*".

IV. Conclusion

We outlined a comprehensive architecture based approach that (Fig. 1) that emphasises all of specific properties and interrelationship of WIS. The paper presented the static and dynamic side of WIS, highlighted the relevant models and modelling approaches. To arrange of all emerged models and views into a disciplined modelling and design approach we have selected the axiomatic design as a design theoretic framework. The refinement of modelling through the design hierarchy fits to the axiomatic design philosophy. The resolution of inconsistency between aspects of WIS represented in the columns can be done by the axiomatic design approach. The contradictions, mismatches and defects of model can be spotted through the Functional Requirements and Design Parameters including the security, identity and access rights of end-users.

The proposed framework provides a comprehensive and general approach for handling the WIS related issues yielding an opportunity for operationalizing the methodology.

Acknowledgment

We are very thankful to Prof. András Benczúr for raising the research issue and helpful discussions.

References

1. O. Noran "A systematic evaluation of the C4ISR AF Using ISO 15704 Annex A (GERAM)". *Computers in Industry*, 56, 407--427, 2005.
2. O. Noran "A mapping of individual architecture frameworks (GRAI, PERA, C4ISR, CIMOSA, Zachman, ARIS) onto GERAM", in: P. Bernus, et al. (Eds.), *Handbook on Enterprise Architecture*, pp. 65--210, Springer-Verlag, Heidelberg, 2003.
3. S. Wang "Toward a general model for web-based information systems", *International Journal of Information Management*, 21, 385--396, 2001
4. D. H. Park, Ch. S. Yoo, Kim, Y. S. Yeom, S. J. "Object modeling for mapping XML document represented in XML-GDM to UML Class Diagram", Lecture Notes in Computer Science, 2006, Volume 3984, *Computational Science and Its Applications* - ICCSA 2006, Pages 958-967
5. E. Köppen, G. Neumann "Active hypertext for distributed web applications", in: Proceedings of The Eighth IEEE International Workshops on *Enabling Technologies: Infrastructure for Collaborative Enterprises* (WET-ICE'99), pp. 297—302, 1999.
6. P. Atzeni, P. Merialdo, Mecca, G., Data-intensive web sites: design and maintenance, *World Wide Web*, 4, pp. 21–47, 2001.
7. J.A. Zachman A Framework for Information Systems Architecture, *IBM Systems Journal Volume*, 26, No. 3, pp. 276--292, 1987.
8. Open Group TOGAF: *The Open Group Architecture Framework, TOGAF®* Version 9, http://www.opengroup.org/togaf/ , 2010.
9. OASIS *A reference model for service-oriented architecture*, White Paper, Service-Oriented Architecture Reference Model Technical Committee, Organization for the Advancement of Structured Information Standards, Billerica, MA, February, 2006.
10. G. Rossi, D. Schwabe, F. Lyardet, "Web application models are more than conceptual models", in: P. Chen et al. (Ed.), *Advances in*

Conceptual Modeling, LNCS, vol. 1727, pp. 239—252, Springer-Verlag, Berlin 1999.

11. Cutter Consortium, *Poor Project Management Number-one Problem of Outsourced E-projects*, Cutter Research Briefs, November, 2000, http://www.cutter.com/research/2000/crb001107.html .
12. St. Spewak, St. Hill *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*, Wiley-QED, 1992.
13. Federal Enterprise Architecture Framework, version 1.1., September 1999, http://www.cio.gov/documents/fedarch1.pdf 2010
14. Department of Defense, *Joint Technical Architecture*, July 2002. http://www.acq.osd.mil/osjtf/pdf/jta-vol-I.pdf 2010
15. US Department of the Treasury Chief Information Officer Council, *Treasury Enterprise Architecture Framework*, http://www.eaframeworks.com/TEAF/teaf.doc , 2010
16. B. Boar *Constructing Blueprints for Enterprise IT Architecture,* John Wiley & Sons, 1999.
17. *ITIL V3 Foundation Handbook: Pocketbook from the Official Publisher of ITIL* - Pack of 10, http://www.best-management-practice.com/Publications-Library/IT-Service-Management-ITIL/ITIL-Version-3/ITIL-V3-Foundation-Handbook/?DI=616573&CLICKID=002259 , 2009.
18. M. Schrefl, E. Kapsammer, B. Pröll, W. Retschitzegger "Self-maintaining web pages—an overview", in: Proceedings of the 12th *Australasian Database Conference* (ADC 2001), IEEE Computer Society, 2001.
19. S. Murugesan, "Web application development: challenges and the role" In: G. Rossi, O. Pastor, D. Schwabe, L. Olsina (eds.) *Web Engineering: Modelling and Implementing Web Applications*, pp. 7—32, Springer-Verlag, London, 2008.
20. T. DeMarco, *Structured Analysis and System Specification*, Prentice Hall, 1979.
21. E. Downs, P. Clare, I. Coe, Structured Systems Analysis and Design Method, Application and Context, Second Edition, Prentice Hall International (UK) Ltd., New York, London, 1992.
22. M. Eva, *SSADM Version 4: A user's guide*, McGraw-Hill, 1992.
23. P. P. Chen „The Entity-Relationship Model: Towards a Unified View of Data", *ACM Transactions on Database Systems*, Vol. 1, No. 1, March 1976, pp 9--36, 1976.
24. M. Bernauer, M. Schrefl "Self-maintaining web pages: from theory to practice", *Data & Knowledge Engineering* 48 , 39--73 2004.
25. C.-M. Chiua, M. Bieber "A dynamically mapped open hypermedia system framework for integrating information systems", *Information and Software Technology* 43, 75--86 2001.
26. C.-K. Nama, G.-S. Jang, J.-H. Ba "An XML-based active document for intelligent web applications", *Expert Systems with Applications*, 25, 165--176, 2003.
27. N.P. Suh *Axiomatic Design: Advantages and Applications*. Oxford University Press, New York, 2001.
28. S.J. Kim, N.P. Suh, S.-K. Kim "Design of software systems based on axiomatic design", *Robotics & Computer-Integrated Manufacturing*, 3, pp. 149--162, 1992.
29. J. Marini, *The Document Object Model: Processing Structured Documents.* McGraw-Hill, 2002.

TABLE 1.
A MAPPING SCHEMATICALLY BETWEEN ZACHMAN ARCHITECTURE AND SOA'S COMPONENT

Aspects / Perspectives	*what*	*how*	*where*	*who*	*when*	*why*	
Contextual	Fact, business data	Business Service	Chain of Business Process, Workflow	Business entity, function	Chain of Business Process, Workflow	Business goal	Scope
Conceptual	Underlying Conceptual data model	Service	Service composition	Actor, Role	Business Process Model	Business Objective	Enterprise Model
Logical	Class hierarchy, Logical Data Model	Service Component	Hierarchy of Service Component	User role, service component	BPEL, BPMN, Orchestration	Business Rule	System Model
Physical	Object hierarchy, Data model	Service Component	Hierarchy of Service Component	Component, Object	Choreography	Rule Design	Technical Model
Detail	Data in DBMS	Service Component	Hierarchy of Service Component	Component, Object	Choreography, Security architecture	Rule specification	Components
Functioning Enterprise	Data	Function	Network	Organization	Schedule	Strategy	

TABLE 2.
ZACHMAN ARCHITECTURE'S RELATIONSHIP TO SOA AND WIS

J. A. Zachman S. H. Spewak	Entities= *what* Data Architecture	Activities= *how* Applications Architecture	Locations= *where* Technology Architecture	People= *who*	Time= *when*	Motivation =*why*	
Planner Objectives/Scope (Contextual)	List of Business Objects *Class*=Class of Business Thing	List of Business Processes Process=Class of Business Process	List of Business Locations Node=Major Business Location	List of Organizations important to the Business People=Major Organizations	List of Events Significant to the Bus. Time=Major Bus. Events	List of Bus. Goals/Strategies Ends/Means=Maj. Bus. Goals/Crit, Suc. Factor	Scope
Owner Enterprise Model (Conceptual)	Semantic Model *Object Class*=Business Entity *Association*=Bus. Relationship	Business Process Model Proc.=Bus. Process, *Web Services* I/O=Bus. Resources, *Documents*	Business Logistics System Node=Busin. Location Link=Business Linkage	Work Flow Model People = Organization Unit Work=Work Product, *Documents*	Master Schedule Time=Bus. Event Cycle=Bus. Cycle	Business Plan Ends=Bus. Objective Means=Bus. Strategy	Enterprise Model
Designer Information Systems Model (Logical)	Logical Data Model *Ent.*=Data Entity *Reln*=Data Relationship	Application Architecture Proc.=Application Function, *Web Services*, *Method* of Object Class, I/O=User Views, *Semi-structured documents*	System Geographic Deployment Architecture e.g. Distributed System Arch. Node=I/S Service. (Processor, Storage, *Logical Application Component.* etc.) Link=Relationship between Logical Appl. Comp.	Human Interface Architecture People=Role Work=Deliverable, *Semi-structured documents*	Processing Stucture Time=System Event, *Orchestration* Cycle=Processing Cycle	Business Rules Ends=Structural Assertion, Means=Action Assertion,	System Model
Builder Technology Model (Physical)	Physical Data Model *Ent.*=Segment/Table/etc *Reln*=Pointer/Key/etc	System Design Proc.= *I/S Services* I/O=Data Elements/Sets, *XML / HTML documents*	System Architecture/Technology Architecture Physical Application Comp. Node=Hardware/Systems Software Link=Line Specifications	Presentation Architecture People=Screen Format, *HTML / XML interface* Work=User	Control Structure Time=Execute, *Choreography* Cycle=Component Cycle	Rule Design Ends=Condition Means=Action	Technical Model
Subcontractor Detailed Specifications (Out-of-context)	Data Definition Repository *Ent.*=Field *Reln*=Address	Programs Supporting Software Components Proc.=Language Statement I/O=Data Item, XML Field	Network Architecture Node=Address Link=Protocol	Security Architecture People=*Identity*, *Authentication*, *Authorization*, Work=Job	Timing Definition Time=Interrupt Cycle=Machine Cycle	Rule Specification Ends=Sub-condition Means=Step	Compo-nents
Functioning Enterprise	Data	Function	Network	Organization	Schedule	Strategy	

TABLE 3.

A SCHEMATICAL MAPPING BETWEEN ZACHMAN ARCHITECTURE AND CNS, FRS, DPS, PVS

Aspects / Perspectives	*what*	*how*	*where*	*who*	*when*	*why*	
Contextual	**CN**	**CN**	**CN**	**CN**	**CN**	**CN**	Scope
Conceptual	**FR**	**FR**	**FR**	**FR**	**FR**	**FR**	Enterprise Model
Logical	**FR /DP**	**FR /DP**	**FR /DP**	**FR /DP**	**FR /DP**	**FR /DP**	System Model
Physical	**FR/DP /PV**	**FR/DP/PV**	**FR/DP /PV**	**FR/DP/PV**	**FR/DP/PV**	**FR/DP/PV**	Technical Model
Detail	**PV**	**PV**	**PV**	**PV**	**PV**	**PV**	Components
Functioning Enterprise	Data	Function	Network	Organization	Schedule	Strategy	

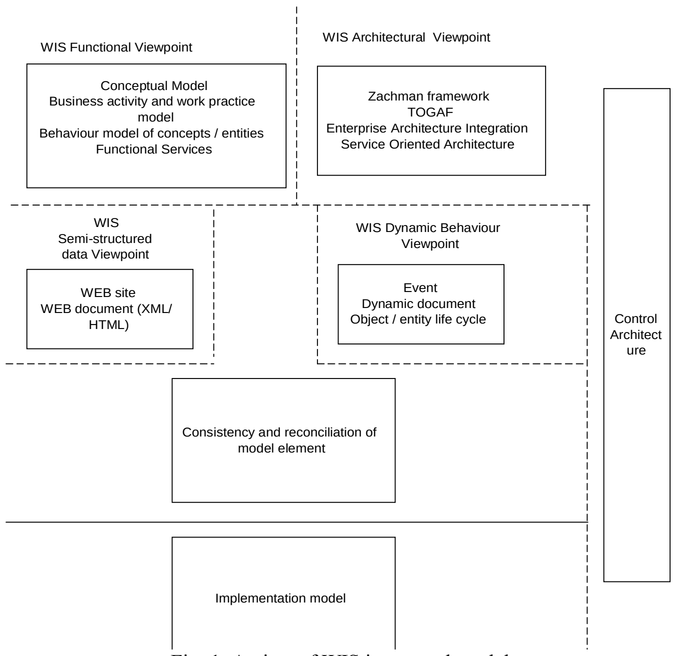
WIS Functional Viewpoint
Conceptual Model
Business activity and work practice model
Behaviour model of concepts / entities
Functional Services
WIS Architectural Viewpoint
Zachman framework
TOGAF
Enterprise Architecture Integration
Service Oriented Architecture
WIS Semi-structured data Viewpoint
WEB site
WEB document (XML/ HTML)
WIS Dynamic Behaviour Viewpoint
Event
Dynamic document
Object / entity life cycle
Control Architecture
Consistency and reconciliation of model element
Implementation model

Fig. 1. A view of WIS integrated model