Enterprise Architecture

An overview

Contents

Articles

Introduction

Enterprise architecture

Enterprise architecture (EA) is the process of translating business vision and strategy into effective enterprise change by creating, communicating and improving the key requirements, principles and models that describe the enterprise's future state and enable its evolution.[1]

Practitioners of EA call themselves *enterprise architects*. An enterprise architect is a person responsible for performing this complex analysis of business structure and processes and is often called upon to draw conclusions from the information collected. By producing this understanding, architects are attempting to address the goals of Enterprise Architecture: Effectiveness, Efficiency, Agility, and Durability.[2]

Definition

Enterprise architecture is an ongoing business function that helps an 'enterprise' figure out how to execute best the strategies that drive its development. The MIT Center for Information Systems Research (MIT CISR) defines enterprise architecture as the specific aspects of a business that are under examination:

> Enterprise architecture is the organizing logic for business processes and IT infrastructure reflecting the integration and standardization requirements of the company's operating model. The operating model is the desired state of business process integration and business process standardization for delivering goods and services to customers.[3]

The United States Government classifies enterprise architecture as an Information Technology function, and defines the term not as the process of examining the enterprise, but rather the documented results of that examination. Specifically, US Code Title 44, Chapter 36, defines it as a 'strategic information base' that defines the mission of an agency and describes the technology and information needed to perform that mission, along with descriptions of how the architecture of the organization should be changed in order to respond to changes in the mission.[4]

Scope

The term *enterprise* is used because it is generally applicable in many circumstances, including

- Public or private sector organizations
- An entire business or corporation
- A part of a larger enterprise (such as a business unit)
- A conglomerate of several organizations, such as a joint venture or partnership
- A multiply outsourced business operation
- Many collaborating public and/or private organizations in multiple countries

The term enterprise includes the whole complex, socio-technical system,[5] including:

- people
- information
- technology
- business (e.g. operations)

Defining the boundary or scope of the enterprise to be described is an important first step in creating the enterprise architecture. *Enterprise* as used in enterprise architecture generally means more than the information systems employed by an organization.[6] A pragmatic enterprise architecture provides a context and a scope. The context

encompasses the (people), organizations, systems and technology out of scope that have relationships with the organisation(s), systems and technology in the scope. In practice, the architect is responsible for the articulation of the scope in the context, engineers are responsible for the details of the scope (just as in the building practice). The architect remains responsible for the work of the engineers, and the implementing contractors thereafter.

Developing an Enterprise Level Architectural Description

Paramount to the enterprise architecture is the identification of the sponsor, his/her mission, vision and strategy and the governance framework to define all roles, responsibilities and relationships involved in the anticipated transition.

As the purpose of architecture is: "INSIGHT, TO DECIDE, FOR ALL STAKEHOLDERS", enterprise architects work very closely with the enterprise sponsor and key stakeholders, internal and external to the enterprise. The architect understands the enterprise mission, vision and strategy and the sponsor's ideas about the approach. The architect articulates the existing enterprise infrastructure value-chain: market, business, systems and technology. Architects present and discuss the technology, systems, business and market options to fulfill the enterprise mission.

Insight is improved by using the 'solution architecture' which is, relative to the decisions ahead, a specific blend of technology, systems, business and market options. Together with the sponsor and the main stakeholders, they make informed choices about the options. For large transitions, architectural decisions are supported by proofs-of-concept and/or business pilots.

Enterprise architects use various methods and tools to capture the structure and dynamics of an enterprise. In doing so, they produce taxonomies, diagrams, documents and models, together called *artifacts*. These artifacts describe the logical organization of business functions, business capabilities, business processes, people, information resources, business systems, software applications, computing capabilities, information exchange and communications infrastructure within the enterprise.

A collection of these artifacts, sufficiently complete to describe the enterprise in useful ways, is considered by EA practitioners an 'enterprise' level architectural description, or enterprise architecture, for short. The UK National Computing Centre EA best practice guidance[7] states

> Normally an EA takes the form of a comprehensive set of cohesive models that describe the structure and functions of an enterprise.
>
> and continues
>
> The individual models in an EA are arranged in a logical manner that provides an ever-increasing level of detail about the enterprise: its objectives and goals; its processes and organization; its systems and data; the technology used and any other relevant spheres of interest.

This is the definition of enterprise architecture implicit in several EA frameworks including the popular TOGAF architectural framework.

An enterprise architecture framework bundles tools, techniques, artifact descriptions, process models, reference models and guidance used by architects in the production of enterprise-specific architectural description. Several enterprise architecture frameworks break down the practice of enterprise architecture into a number of practice areas or *domains*.

See the related articles on enterprise architecture frameworks and domains for further information.

In 1992, Steven Spewak described a process for creating an enterprise architecture that is widely used in educational courses.[8]

Using an enterprise architecture

Describing the architecture of an enterprise aims primarily to improve the effectiveness or efficiency of the business itself. This includes innovations in the structure of an organization, the centralization or federation of business processes, the quality and timeliness of business information, or ensuring that money spent on information technology (IT) can be justified.[2]

One method of using this information to improve the functioning of a business, as described in the TOGAF architectural framework, involves developing an "architectural vision": a description of the business that represents a "target" or "future state" goal. Once this vision is well understood, a set of intermediate steps are created that illustrate the process of changing from the present situation to the target. These intermediate steps are called "transitional architectures" by TOGAF.[9] Similar methods have been described in other enterprise architecture frameworks.

Benefits of enterprise architecture

As new technologies arise and are implemented, the benefits of enterprise architecture continue to grow. Enterprise architecture defines what an organization does; who performs individual functions within the organization; how the organizational functions are performed; and how data is used and stored. IT costs are reduced and responsiveness with IT systems is improved. However, to be successful, continual development and periodic maintenance of the enterprise architecture is essential. Building an enterprise architecture could take considerable time and proper planning essential, including phasing the project in slowly, prior to implementation. If the enterprise architecture is not kept up to date, the aforementioned benefits will become useless.

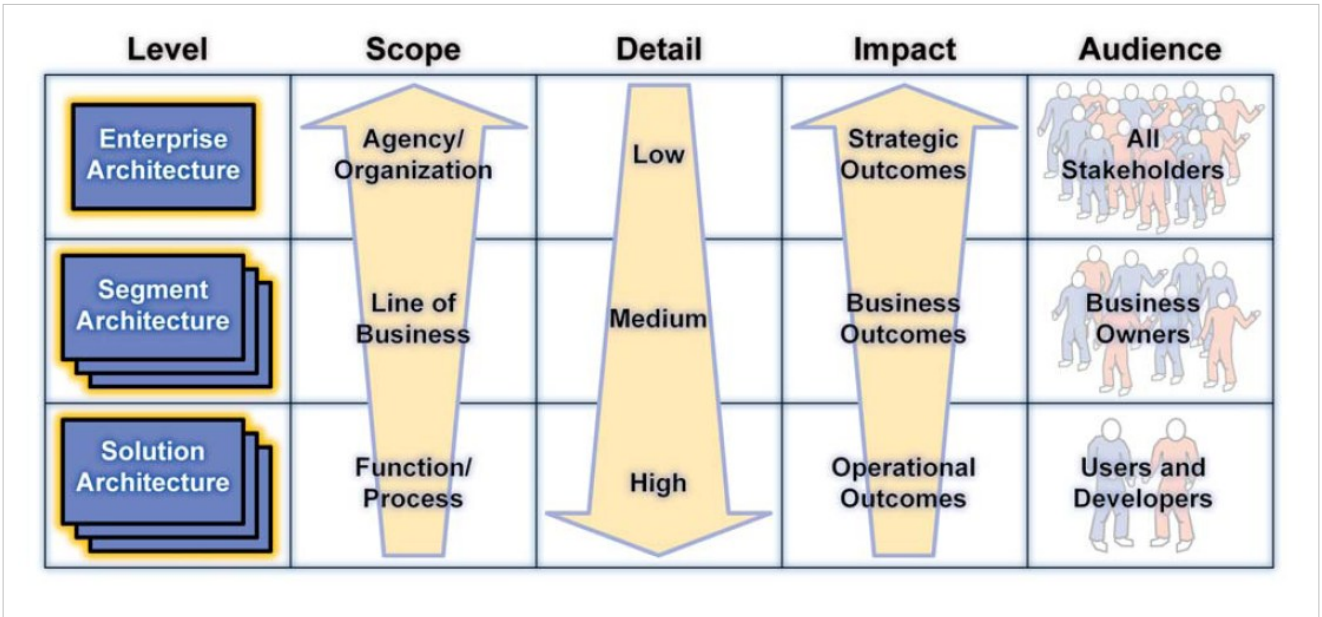
The growing use of enterprise architecture

Documenting the architecture of enterprises is done within the U.S. Federal Government[10] in the context of the Capital Planning and Investment Control (CPIC) process. The Federal Enterprise Architecture (FEA) reference models guides federal agencies in the development of their architectures.[11] Companies such as Independence Blue Cross, Intel, Volkswagen AG[12] and InterContinental Hotels Group[13] also use enterprise architecture to improve their business architectures as well as to improve business performance and productivity.

Relationship to other disciplines

Enterprise architecture is a key component of the information technology governance process in many organizations, which have implemented a formal enterprise architecture process as part of their IT management strategy. While this may imply that enterprise architecture is closely tied to IT, it should be viewed in the broader context of business optimization in that it addresses business architecture, performance management and process architecture as well as more technical subjects. Depending on the organization, enterprise architecture teams may also be responsible for some aspects of performance engineering, IT portfolio management and metadata management. Recently, protagonists like Gartner and Forrester have stressed the important relationship of Enterprise Architecture with emerging holistic design practices such as Design Thinking and User Experience Design.[14]

The following image from the 2006 FEA Practice Guidance of US OMB sheds light on the relationship between enterprise architecture and segment (BPR) or Solution architectures. (This figure demonstrates that software architecture is truly a solution architecture discipline, for example.)



Activities such as software architecture, network architecture, and database architecture are partial contributions to a solution architecture.

Published examples

It is uncommon for a commercial organization to publish rich detail from their enterprise architecture descriptions. Doing so can provide competitors information on weaknesses and organizational flaws that could hinder the company's market position. However, many government agencies around the world have begun to publish the architectural descriptions that they have developed. Good examples include the US Department of the Interior, US Department of Defense Business Enterprise Architecture, or the 2008 BEAv5.0 version.

Academic qualifications

Enterprise Architecture was included in the Association for Computing Machinery (ACM) and Association for Information Systems (AIS)'s Curriculum for Information Systems as one of the 6 core courses.[15] There are several universities that offer enterprise architecture as a fourth year level course or part of a master's syllabus. The Center for Enterprise Architecture [16] at the Penn State University is one of these institutions that offer EA courses. It is also offered within the Masters program in Computer Science at The University of Chicago. In 2010 researchers at the Meraka Institute, Council for Scientific and Industrial Research, in South Africa organized a workshop and invited staff from computing departments in South African higher education institutions. The purpose was to investigate the current status of EA offerings in South Africa. A report was compiled and is available for download at the Meraka Institute.[17]

References

[1] Definition of Enterprise Architecture, Gartner (http://www.gartner.com/technology/it-glossary/enterprise-architecture.jsp)

[2] Pragmatic Enterprise Architecture Foundation, PEAF Foundation - Vision (http://www.pragmaticea.com/display-doc.asp?DocName=peaf-foundation-vision)

[3] MIT Center for Information Systems Research, Peter Weill, Director, as presented at the Sixth e-Business Conference, Barcelona Spain, 27 March 2007, (http://www.iese.edu/en/files/6_29338.pdf)

[4] U.S.C. Title 44, Chap. 36, § 3601 (http://us-code.vlex.com/vid/sec-definitions-19256361)

[5] Giachetti, R.E., Design of Enterprise Systems, Theory, Architecture, and Methods, CRC Press, Boca Raton, FL, 2010.

[6] (http://enterprisearchitecture.nih.gov/About/What/enterprisearchitecture.nih.gov)

[7] Jarvis, R, Enterprise Architecture: Understanding the Bigger Picture - A Best Practice Guide for Decision Makers in IT, The UK National Computing Centre, Manchester, UK

[8] Spewak, Steven H. and Hill, Steven C. , Enterprise Architecture Planning - Developing a Blueprint for Data Applications and Technology,(1992), John Wiley

[9] The Open Group, TOGAF standard, http://www.opengroup.org/togaf/

[10] Federal Government agency success stories, (2010), whitehouse.gov (http://www.whitehouse.gov/omb/E-Gov/ea_success.aspx)

[11] FEA Practice Guidance Federal Enterprise Architecture Program Management Office OMB, (2007), whitehouse.gov (http://www.whitehouse.gov/sites/default/files/omb/assets/fea_docs/FEA_Practice_Guidance_Nov_2007.pdf)

[12] "Volkswagen of America: Managing IT Priorities," Harvard Business Review, October 5, 2005, Robert D. Austin, Warren Ritchie, Greggory Garrett

[13] ihg.com (http://www.ihg.com)

[14] Leslie Owens, Forrester Blogs - Who Owns Information Architecture? All Of Us., (2010), blogs.forrester.com (http://blogs.forrester.com/information_management/2010/02/who-owns-information-architecture-all-of-us.html)

[15] ACM and AIS Curriculum for Information Systems acm.org (http://www.acm.org/education/curricula/IS 2010 ACM final.pdf)

[16] Center for Enterprise Architecture, Penn State University, ea.ist.psu.edu (http://ea.ist.psu.edu/)

[17] hufee.meraka.org.za (http://hufee.meraka.org.za/Hufeesite/mekes-projects)

External links

- Professional Practice Guide for Enterprise Architects (http://caeap.org/IndustryArtifacts.aspx)

University and college programs

- Pennsylvania State University (http://ea.ist.psu.edu)
- Carnegie Mellon (http://execed.isri.cmu.edu/elearning/enterprise-architecture/index.html)
- National University (http://www.nu.edu/OurPrograms/SchoolOfEngineeringAndTechnology/AppliedEngineering/Programs/720-810.html)
- Kent State University (http://www.kent.edu/dsci/enterprisearchitecture/index.cfm)
- Griffith University (http://www17.griffith.edu.au/cis/p_cat/admission.asp?ProgCode=5493&Type=apply)
- Royal Melbourne Institute of Technology (http://www.rmit.edu.au/browse/Our Organisation/Science Engineering and Health/Schools/Computer Science and IT/Programs and Courses/Postgraduate/MC152 Master of Technology (Enterprise Architecture))
- Temple University, Fox School of Business (http://community.mis.temple.edu/mis2501sec001s12/)
- University of Utrecht, Dept of Information and Computing Sciences (http://www.cs.uu.nl/education/vak.php?vak=INFOEAR)

Enterprise Architect

Enterprise architect

Enterprise architects are practitioners of enterprise architecture; an information technology management discipline that operates within organizations.

Role

Enterprise architects work with stakeholders, both leadership and subject matter experts, to build a holistic view of the organization's strategy, processes, information, and information technology assets. The role of the enterprise architect is to take this knowledge and ensure that the business and IT are in alignment. The enterprise architect links the business mission, strategy, and processes of an organization to its IT strategy, and documents this using multiple architectural models or views that show how the current and future needs of an organization will be met in an efficient, sustainable, agile, and adaptable manner.

Enterprise architects operate across organizational and computing "silos" to drive common approaches and expose information assets and processes across the enterprise. Their goal is to deliver an architecture that supports the most efficient and secure IT environment meeting a company's business needs.

Enterprise architects are like city planners, providing the roadmaps and regulations that a city uses to manage its growth and provide services to its citizens. In this analogy, it is possible to differentiate the role of the system architect, who plans one or more buildings; software architects, who are responsible for something analogous to the HVAC (Heating, Ventilation and Air Conditioning) within the building; network architects, who are responsible for something like the plumbing within the building, and the water and sewer infrastructure between buildings or parts of a city. The enterprise architect however, like a city planner, both frames the city-wide design, and choreographs other activities into the larger plan.[1]

Delivered successfully, an enterprise architecture has the potential to allow both the Business and IT strategies to enable and drive each other. Therefore, effective enterprise architecture may be regarded as one of the key means to achieving competitive advantage through information technology.

Responsibilities

- Alignment of IT strategy and planning with company's business goals.
- Optimization of information management approaches through an understanding of evolving business needs and technology capabilities.
- Long-term strategic responsibility for the company's IT systems.
- Promotion of shared infrastructure and applications to reduce costs and improve information flows. Ensure that projects do not duplicate functionality or diverge from each other and business and IT strategies.
- Work with solutions architect(s) to provide a consensus based enterprise solution that is scalable, adaptable and in synchronization with ever changing business needs.
- Management of the risks associated with information and IT assets through appropriate standards and security policies.
- Direct or indirect involvement in the development of policies, standards and guidelines that direct the selection, development, implementation and use of Information Technology within the enterprise.
- Build employee knowledge and skills in specific areas of expertise.

Skills and knowledge

- Systems thinking - the ability to see how parts interact with the whole (big picture thinking)
- Knowledge of the business for which the enterprise architecture is being developed
- Interpersonal and leadership skills - servant leadership, collaboration, facilitation, and negotiation skills
- Communication skills, both written and spoken
- Ability to explain complex technical issues in a way that non-technical people may understand
- Knowledge of IT governance and operations
- Comprehensive knowledge of hardware, software, application, and systems engineering
- Project and program management planning and organizational skills
- Knowledge of financial modeling as it pertains to IT investment
- Customer service orientation
- Time management and prioritization

References

[1] Rosen, Mike *Ten Key Skills Architects Must Have to Deliver Value* (http://www.technologytransfer.eu/article/78/2009/9/Ten_Things_an_Architect_Does_to_Add_Value.html), November 2008

External links

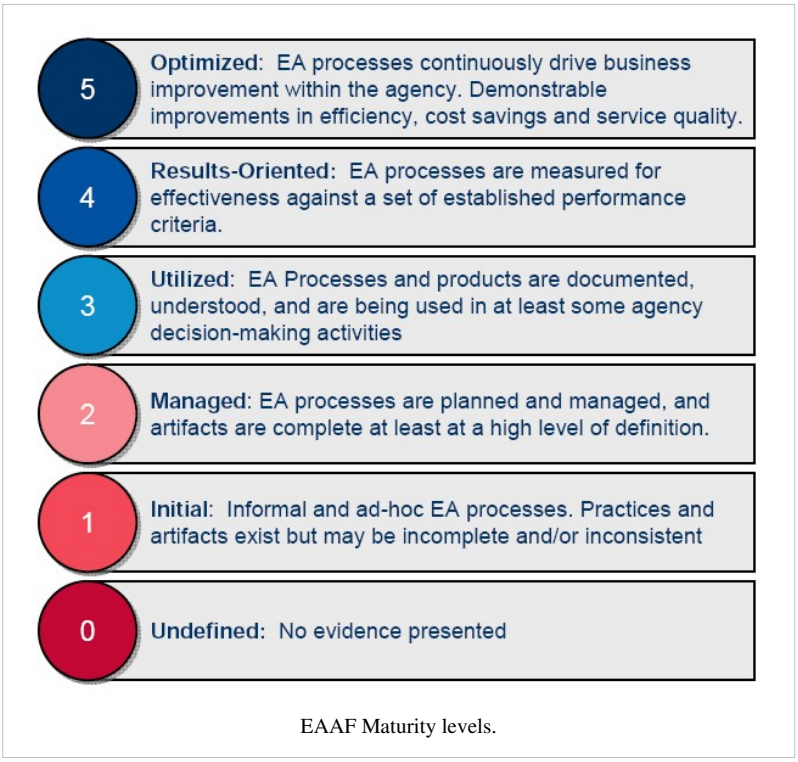
- IBM Developer Works: Article on SOA user roles (http://www-128.ibm.com/developerworks/webservices/library/ws-soa-progmodel10/)
- Daljit Banger Rise of the Enterprise Architect (http://www.bcs.org/server.php?show=ConWebDoc.3333), British Computer Society, March 2006

Enterprise Architecture

Enterprise Architecture Assessment Framework

The **Enterprise Architecture Assessment Framework** (**EAAF**) is created by the US Federal government Office of Management and Budget (OBM) to allow federal agencies to assess and report their enterprise architecture activity and maturity[1], and to advance the use of enterprise architecture in the federal government.[2]

The version 2.2 of the framework was released in October 2007[3], and version in June 2009.



EAAF Maturity levels.

Overview

The OMB Enterprise Architecture Assessment Framework (the Framework) helps OMB and the agencies assess the capability of enterprise architecture programs to guide IT investments.[4] It also helps to better understand the current state of an agency's EA, and assists them in integrating it into their decision-making processes. By applying the assessment themselves, agencies can identify strengths and weaknesses within their programs and adjust them accordingly.

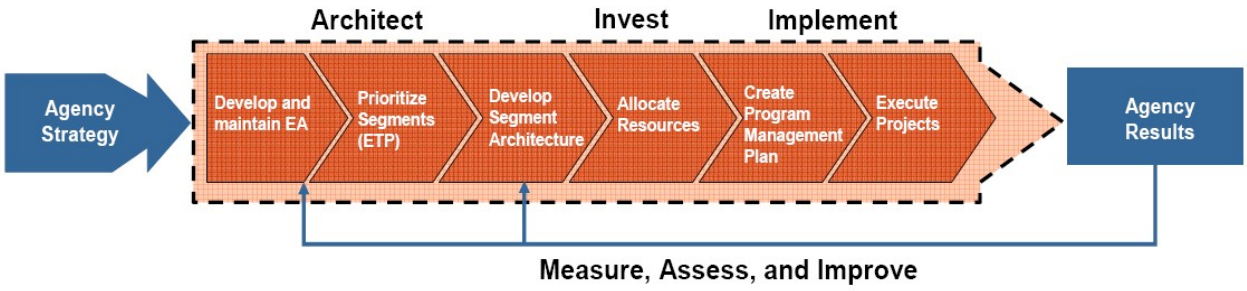
Enterprise Architecture Assessment Framework (EAAF) version 3.1 identifies the measurement areas and criteria by which agencies are expected to use the EA to drive performance improvements that result in the following outcomes[1]:

- Closing agency performance gaps identified via coordinated agency strategic planning and performance management activities;
- Saving money and avoiding cost through collaboration and reuse, productivity enhancements, and elimination of redundancy;
- Strengthening the quality of agency investment portfolios by improving security, inter-operability, reliability, availability, solution development and service delivery time, and overall end-user performance;
- Improving the quality, availability and sharing of data and information government-wide; and
- Increasing the transparency of government operations by increasing the capacity for citizen participation and cross-governmental collaboration.

While agencies have clearly demonstrated a degree of maturity and competency in developing and using their EAs, EAAF seeks to advance the practice of EA, particularly through the development and use of agency segment architectures, aimed at driving the kinds of government-wide outcomes.[5]

Performance Improvement Lifecycle

Government agencies are continually assessing current performance, identifying opportunities for improvement, and translating them into specific actions. Enterprise architecture is an integrated management practice that maximizes the use of an agency's resources to achieve their goals. Architecture describes the pathway from strategic goals and objectives, through investments, to measurable performance improvements for the entire enterprise or a portion.[5]



Information and IT-Enabled Performance Improvement Lifecycle

Continuous performance improvement is the principal driver connecting EA program staff with key business stakeholders across each phase of the Performance Improvement Lifecycle. Agency Chief Architects and EA program staff:[5]

- identify and prioritize enterprise segments and opportunities to improve mission performance, linked to agency goals and objectives;
- plan a course of action to close performance gaps, using common or shared information assets and information technology assets;
- allocate agency resources supporting program management and project execution;
- measure and assess performance to verify and report results; and
- assess feedback on program performance to enhance architecture, investment and implementation decisions.

Opportunities to improve mission performance are prioritized in terms of their relative value to the agency's strategic goals and objectives in the enterprise transition plan (ETP) and segment architecture.[5]

Architect

Enterprise architecture describes the current (baseline) and future (target) states of the agency, and the plan to transition from the current to the future state, with a focus on agency strategy, program performance improvements and information technology investments. Agency EAs are organized by segments – core mission areas (e.g., homeland security, health), business service (e.g., financial management, human resources), and enterprise services (e.g., Information Sharing). Segments are defined using the Federal Enterprise Architecture (FEA) reference models .[5]

Invest

Performance improvement opportunities identified during the "Architect" process are ideally addressed through an agency portfolio of IT investments. This step defines the implementation and funding strategy for individual initiatives in the Enterprise Transition Plan (ETP) and described in the segment architectures. Program management plans are created to implement the individual solutions.[5]

Implement

Projects are executed and tracked throughout the system development life cycle (SDLC). Achievement of the program / project plan within acceptable variance for schedule and budget is measured and reported through Earned Value Management (EVM) process.[5]

Measure, assess and improve

Information and information technology, as critical enablers of program performance improvements, must be assessed and evaluated in the context of agency missions and outcome-oriented results defined in the enterprise-wide performance architecture.[5]

Maturity	Completion	Use	Results
INCREASING LEVELS OF MATURITY	• Average score equal to or greater than 4 in the "Completion" capability area	• Average score equal to or greater than 4 in the "Use" capability area	• Average score equal to or greater than 4 in the "Results" capability area
	• Score equal to or greater than 3 in the "Completion" capability area	• Score equal to or greater than 3 in the "Use" capability area	• Score equal to or greater than 3 in the "Results" capability area
	• Score less than 3 in the "Completion" capability area	• Score less than 3 in the "Use" capability area	• Score less than 3 in the "Results" capability area

EAAF assessment table describing how agency EAs will be assessed.

Performance improvement plans and priorities, including those previously gathered under the PART and Performance Assessment Report (PAR) programs, are reflected in the agency EA, particularly the performance architecture and ETP. Performance metrics previously gathered are used to evaluate the results in agency performance improvement plans, identifying a program's strengths and weaknesses and addressing ways to improve the program performance.[5]

Agency submission data quality

OMB collects a significant amount of IT investment data and other related data from executive agencies during each phase of Performance Improvement Lifecycle. OMB uses the information for development of an IT investment portfolio as a part of the President's budget request to Congress.[5]

References

[1] Enterprise Architecture Assessment Framework (http://www.whitehouse.gov/omb/e-gov/eaaf/), Office of Management and Budget, USA.

[2] Pallab Saha (2009) *Advances in Government Enterprise Architecture*. Idea Group Inc (IGI). p.133

[3] OMB (july 2007) *Federal Enterprise Architecture Program EA Assessment Framework 2.2*. (Online copy here (http://www.scribd.com/doc/6846223/OMB-EA-Assessment-Framework-22))

[4] Federal Enterprise Architecture (http://www.whitehouse.gov/omb/e-gov/fea/), Office of Management and Budget, USA.

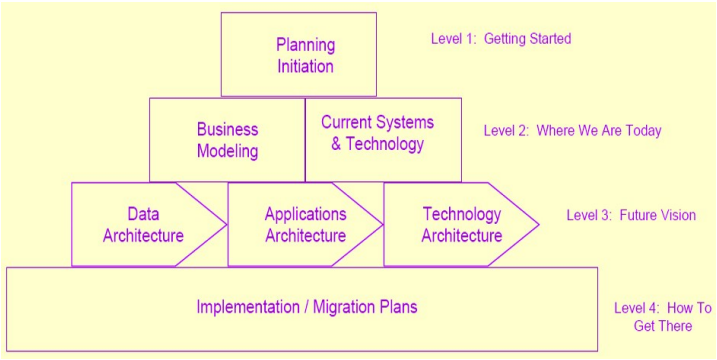
[5] US OBM (2009). Improving Agency Performance Using Information and Information Technology : Enterprise Architecture Assessment Framework v3.1 (http://www.whitehouse.gov/sites/default/files/omb/assets/fea_docs/OMB_EA_Assessment_Framework_v3_1_June_2009.pdf) June 2009

External links

- E-GOV EA Assessment (http://www.whitehouse.gov/omb/e-gov/eaaf/) at whitehouse.gov

Enterprise architecture planning

Enterprise Architecture Planning (EAP) in Enterprise Architecture is the planning process of defining architectures for the use of information in support of the business and the plan for implementing those architectures.[2]



Levels of Enterprise Architecture Planning.[1]

Overview

One of the earlier professional practitioners in the field of system architecture Steven H. Spewak in 1998 defined Enterprise Architecture Planning (EAP) as "the process of defining architectures for the use of information in support of the business and the plan for implementing those architectures." Spewak's approach to EAP is similar to that taken by DOE in that the business mission is the primary driver. That is followed by the data required to satisfy the mission, followed by the applications that are built using that data, and finally by the technology to implement the applications.[1]

This hierarchy of activity is represented in the figure above, in which the layers are implemented in order, from top to bottom. Based on the Business Systems Planning (BSP) approach developed by John Zachman, EAP takes a data-centric approach to architecture planning to provide data quality, access to data, adaptability to changing requirements, data interoperability and sharing, and cost containment. This view counters the more traditional view that applications should be defined before data needs are determined or provided for.[1]

EAP topics

Zachman framework

EAP defines the blueprint for subsequent design and implementation and it places the planning/defining stages into a framework. It does not explain how to define the top two rows of the Zachman Framework in detail but for the sake of the planning exercise, abbreviates the analysis. The Zachman Framework provides the broad context for the description of the architecture layers, while EAP focuses on planning and managing the process of establishing the business alignment of the architectures.[2]

EAP is planning that focuses on the development of matrixes for comparing and analyzing data, applications, and technology. Most important, EAP produces an implementation plan. Within the Federal Enterprise Architecture, EAP will be completed segment enterprise by segment enterprise. The results of these efforts may be of Governmentwide value; therefore, as each segment completes EAP, the results will be published on the ArchitecturePlus web site.[2]

EAP components

Enterprise Architecture Planning model consists of four levels:

- Layer 1 - *getting started* : This layer leads to producing an EAP workplan and stresses the necessity of high-level management commitment to support and resource the subsequent six components (or steps) of the process. It consists of Planning Initiation, which covers in general, decisions on which methodology to use, who should be involved, what other support is required, and what toolset will be used.[2]
- Layer 2 - *where we are today* : This layer provides a baseline for defining the eventual architecture and the long-range migration plan. It consists of:[2]
  - Business process modeling, the compilation of a knowledge base about the business functions and the information used in conducting and supporting the various business processes, and
  - Current Systems and Technology, the definition of current application systems and supporting technology platforms.
- Layer 3 - *the vision of where we want to be* : The arrows delineate the basic definition process flow: data architecture, applications architecture, and technology architecture. It consists of:[2]
  - Data Architecture - Definition of the major kinds of data needed to support the business.
  - Applications Architecture - Definition of the major kinds of applications needed to manage that data and support the business functions.
  - Technology Architecture - Definition of the technology platforms needed to support the applications that manage the data and support the business functions.
- Layer 4 - *how we plan to get there* : This consists of the Implementation / Migration Plans - Definition of the sequence for implementing applications, a schedule for implementation, a cost/benefit analysis, and a clear path for migration.[2]

EAP methodology

The Enterprise Architecture Planning (EAP) methodology is beneficial to understanding the further definition of the Federal Enterprise Architecture Framework at level IV. EAP is a how to approach for creating the top two rows of the Zachman Framework, Planner and Owner. The design of systems begins in the third row, outside the scope of EAP.[2]

EAP focuses on defining what data, applications, and technology architectures are appropriate for and support the overall enterprise. Exhibit 6 shows the seven components (or steps) of EAP for defining these architectures and the related migration plan. The seven components are in the shape of a wedding cake, with each layer representing a different focus of each major task (or step).[2]

Applications

Spewak approach to Federal Enterprise Architecture has helped organizations with modeling, business strategy planning, process improvement, data warehousing, and various support systems designs, data administration standards, object-oriented and information engineering methodologies, and project management.[2]

References

[1] FAA (1998). Federal Information Architecture Initiatives (http://www.faa.gov/niac/pdf/wn18_fia.pdf) Federal Aviation Administration, February 1998.

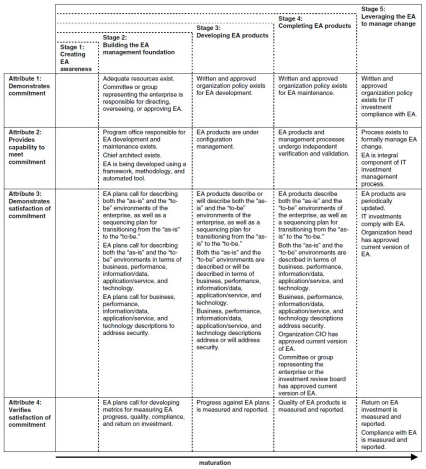
[2] The Chief Information Officers Council (1999). *Federal Enterprise Architecture Framework Version 1.1* (http://www.cio.gov/Documents/fedarch1.pdf) September 1999.

Further reading

- Steven Spewak with Steven C. Hill (1995) *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications, and Technology*. John Wiley & Sons, New York City. 1995.

Enterprise Architecture Management

Enterprise Architecture Management (or **EAM**) in the field of Enterprise architecture describes and structures complex IT systems in terms of their business, application, information, and technical layers, and to reform programs through the planning process as strategic business demands, and as standards and guidelines for the development of local solutions and service offers.



	Stage 1: Creating EA awareness	Stage 2: Building the EA management foundation	Stage 3: Developing EA products	Stage 4: Completing EA products	Stage 5: Leveraging the EA to manage change
Attribute 1: Demonstrates commitment		Adequate resources exist. Committee or group representing the enterprise is responsible for directing, overseeing, or approving EA.	Written and approved organization policy exists for EA development.	Written and approved organization policy exists for EA maintenance.	Written and approved organization policy exists for IT investment compliance with EA.
Attribute 2: Provides capability to meet commitment		Program office responsible for EA development and maintenance exists. Chief architect exists. EA is being developed using a framework, methodology, and automated tool.	EA products are under configuration management.	EA products and management processes undergo independent verification and validation.	Process exists to formally manage EA change. EA is integral component of IT investment management process.
Attribute 3: Demonstrates satisfaction of commitment		EA plans call for describing both the "as-is" and the "to-be" environments of the enterprise, as well as a sequencing plan for transitioning from the "as-is" to the "to-be." EA plans call for describing both the "as-is" and the "to-be" environments in terms of business, performance, information/data, application/service, and technology. EA plans call for business, performance, information/data, application/service, and technology descriptions to address security.	EA products describe or will describe both the "as-is" and the "to-be" environments of the enterprise, as well as a sequencing plan for transitioning from the "as-is" to the "to-be." Both the "as-is" and the "to-be" environments are described or will be described in terms of business, performance, information/data, application/service, and technology. Business, performance, information/data, application/service, and technology descriptions address or will address security.	EA products describe both the "as-is" and the "to-be" environments of the enterprise, as well as a sequencing plan for transitioning from the "as-is" to the "to-be." Both the "as-is" and the "to-be" environments are described in terms of business, performance, information/data, application/service, and technology. Business, performance, information/data, application/service, and technology descriptions address security. Organization CIO has approved current version of EA. Committee or group representing the enterprise or the investment review board has approved current version of EA.	EA products are periodically updated. IT investments comply with EA. Organization head has approved current version of EA.
Attribute 4: Verifies satisfaction of commitment		EA plans call for developing metrics for measuring EA progress, quality, compliance, and return on investment.	Progress against EA plans is measured and reported.	Quality of EA products is measured and reported.	Return on EA investment is measured and reported. Compliance with EA is measured and reported.

maturation

Enterprise Architecture Management Maturity Framework : Maturity Stages, Critical Success Attributes, and Core Elements.[1]

Overview

The fundamental prerequisites for effective EAM are a current, consistent baseline of information about the as-is landscape and an integrated planning process from demand to budget to reach the to-be landscape. The enterprise architecture function also involves reviewing and consolidating detailed architecture decisions and migration plans to identify efficiencies, advance standardization, and align business and IT priorities.

As IT architectural layers, business support processes, and organizational structures become more sophisticated and prone to constant change, EAM will only result in haphazard business and IT alignment if the primary focus is on delivering sets of technically based models. This approach is only helpful insofar as it depicts the enterprise architecture as a snapshot in time, but it offers no reiterative process support to develop architecture solutions and test against different scenarios, benchmarks, and standards as dictated by the ever converging business and IT strategy.

Moreover, a model-centric approach is prohibitively time-intensive to keep updated and leaves too much room for error as changes to the architecture occur unchecked and isolated in the heads of small groups of architecture specialists. Instead, the EAM effort has to bring the highly distributed knowledge of all experts to the table and allow every participant to provide such knowledge and input in the terms that best fit the experience and expectations of the contributing stakeholders.

Application

Successful Enterprise Architecture programs are approached from a management perspective as opposed to a modeling perspective. A new generation of EA Planning tools are emerging that support not only the modeling of the architecture, but also the creation of roll-out and implementation plans for continuous IT improvement over time.

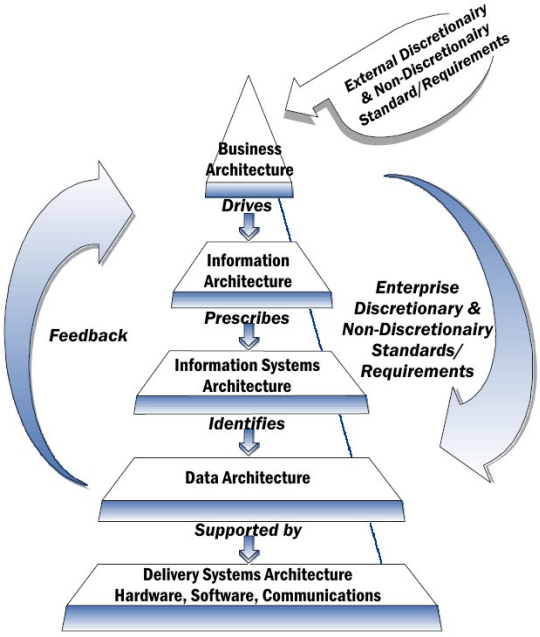
An important aspect of this approach is support of collaboration amongst a wide group of stakeholders from both business and IT including C-level, IT strategists, planning teams, technology implementers, and business analysts, who contribute to the EA management and planning process. In this way EAM supports sustainable business strategy realization.

References

[1] United States General Accounting Office (2003-04). A Framework for Assessing and Improving Enterprise Architecture Management (Version 1.1). April 2003. Retrieved from http://www.gao.gov/new.items/d03584g.pdf.

Enterprise Architecture framework

An **enterprise architecture framework** (EA framework) is a architecture framework which defines how to organize the structure and views associated with an enterprise architecture.



NIST Enterprise Architecture Model initiated in 1989, one of the earliest frameworks for enterprise architecture.[1]

Overview

The three components of the enterprise architecture framework are:[2]

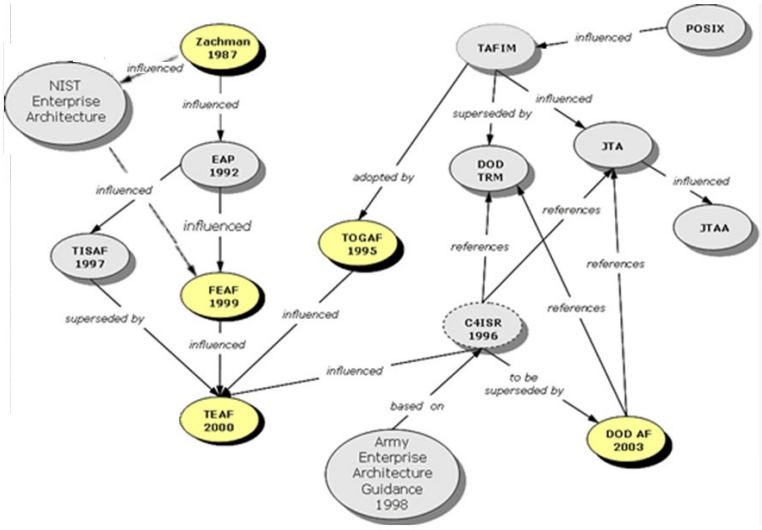
- Views : provide the mechanisms for communicating information about the relationships that are important in the architecture
- Methods : provide the discipline to gather and organize the data and construct the views in a way that helps ensure integrity, accuracy and completeness
- Training/Experience : support the application of method and use of tools

Because the discipline of enterprise engineering and enterprise architecture is so broad, and because enterprises can be large and complex, the models associated with the discipline also tend to be large and complex. To manage this scale and complexity, an architecture framework provides tools and methods that can bring the task into focus and allow valuable artifacts to be produced when they are most needed.

Architecture frameworks are commonly used in IT and information system governance. An organization may wish to mandate that certain models be produced before a system design can be approved. Similarly, they may wish to specify certain views be used in the documentation of procured systems – the U.S. Department of Defense stipulates that specific DoDAF views be provided by equipment suppliers for capital project above a certain value. This discussion is focused for theorical teams who will develop tools and empirical methods of quality achievements.

History

Enterprise architecture started with the Zachman Framework in 1987. Another early implementation of an Enterprise architecture framework was the "Technical Architecture Framework for Information Management" (TAFIM). The first draft of TAFIM was completed in 1991 with the TAFIM Technical Reference Model (TAFIM TRM). This technical reference model wanted to use open systems and new technologies available in the commercial market, to develop a DoD-wide application.[3] The TOGAF TRM was originally derived from the Technical Architecture Framework for Information Management (TAFIM), which in turn was derived from the IEEE model 1003.0[4] or POSIX Open System Environment: a standard "to construct an information processing system, including consumers, system integrators, application developers, system providers, and procurement agencies".[5]



Impression of Enterprise Architecture Frameworks evolution (1987–2003).[2] On the left: The Zachman Framework 1987, NIST Enterprise Architecture 1989, EAP 1992, TISAF 1997, FEAF 1999 and TEAF 2000. On the right: POSIX, TAFIM, JTA, JTAA, TOGAF 1995, DoD TRM and C4ISR 1996, and DoDAF 2003.

In recent years, it has become apparent that a key benefit to be gained from enterprise architecture is the ability to support decision making in changing businesses. Because enterprise architecture brings together business models (e.g. process models, organizational charts, etc.) and technical models (e.g. systems architectures, data models, state diagrams, etc.) it is possible to trace the impact of organizational change on the systems, and also the business impact of changes to the systems.

As this benefit has emerged, many frameworks such as DoDAF, MODAF, or AGATE have adopted a standard meta model which defines the critical architectural elements and the dependencies between them. Applications based on these models can then query the underlying architectural information, providing a simple and strong mechanism for tracing strategies to organizational and technological impacts.

EA framework topics

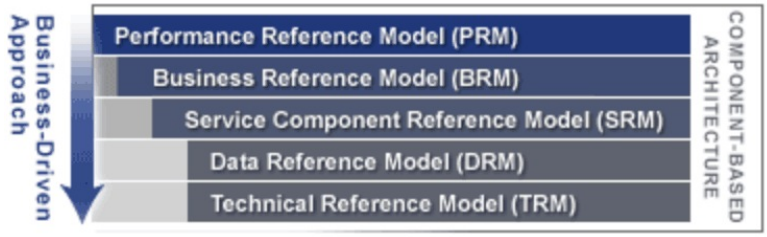
Framework of building codes

Artist impression.[6]

Persons who have ever remodeled their home, know how important building codes, blueprints, and city or county inspections are to successfully complete the project. The architect operates within a "framework" of building codes, preparing blueprints for each phase of the project, from the structural changes to the size and layout of the rooms. Detailed drawings specify plumbing, electrical, and building construction information for the entire structure. Enterprise Architecture works in a similar manner.[6]

An architecture framework for IT affects every aspect of the enterprise. An enterprise architecture framework is similar to building codes that ensure the building is soundly constructed. The IT governance bodies and procedures serve as the city and county inspectors for building improvement projects. Frameworks contain models and standards that will be used to develop IT architecture descriptions. The architecture description is the blueprint.[6]

Architecture domain



Example of the federal enterprise architecture, which has defined five architectural layers.[7]

In the context of the creation of Enterprise Architecture it is common, according to Péter Bernus (2005),[8] to recognise three or four types of architecture, each corresponding to its particular architecture domain. Examples of such domains are:

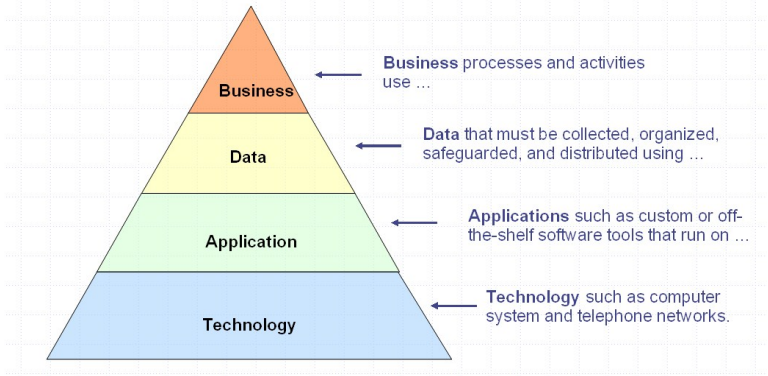
- Business architecture,
- Information systems architecture, often subdivided into
  - Data architecture, and
  - Application architecture,
- and Technical architecture.

Architectural domains are a structuring criterion for a collection of architecture products. They should not be confused with the application domain of the framework as such.[8]

Layers of the enterprise architecture

Contemporary federal guidance suggests thinking about “layers” of the enterprise architecture:[9]

- Business processes and activities
- Data that must be collected, organized, safeguarded, and distributed
- Applications such as custom or off-the-shelf software tools
- Technology such as computer systems and telephone networks

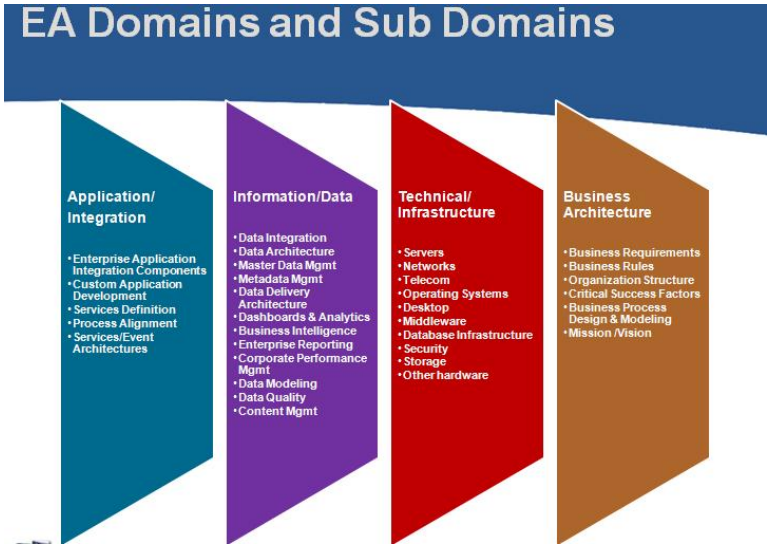


Layers of the enterprise architecture.[9]

The Architecture Domains follow a pattern of decomposition as one goes from top to the bottom of the framework. The ownership can be divided into 4 broad categories: planner's view, owner's view, designer's view and developer's view in this order. All the views are mostly hierarchical in nature. For business view the planner and owner's level is typically called the value chains (which are descriptive by nature). The designer's view of business is also known as the analytical view and there are various standards for modeling this view. One commonly used modeling standard is the Business Process Modeling Notation (BPMN). The designer's view typically represents the execution level which uses standards like Business Process Execution Language (BPEL).

Enterprise architecture domains and subdomains

The application and technology domains (which are not to be confused with business domains) are characterized by domain capabilities and domain services. The capabilities are supported by the services. The application services are also referred to in service-oriented architecture (SOA). The technical services are typically supported by software products.



Enterprise architecture reference architecture with sub domains

The data view starts with the data classes which can be decomposed into data subjects which can be further decomposed into data entities. The basic data model type which is most commonly used is called merda (master entity relationship diagrams assessment, see entity-relationship model). The Class, subject and entity forms a hierarchical view of data. Enterprises may have millions of instances of data entities.

The Enterprise Architecture Reference Traditional Model offers clear distinction between the architecture domains (business, information/data, application/integration and technical/infrastructure). These domains can be further divided into Sub domain disciplines. An example of the EA domain and sub domains is in the image on the right.

Many enterprise architecture teams consist of Individuals with Skills aligned with the Enterprise Architecture Domains and sub-domain disciplines. Here are some examples: enterprise business architect, enterprise

documentational architect, enterprise application architect, enterprise infrastructure architect, enterprise information architect, etc.

An example of the list of reference architecture patterns in the application and information architecture domains are available at Architectural pattern (computer science).

View model

A view model is a framework, which defines the set of views or approaches to be used in systems analysis or systems design or the construction of an enterprise architecture.

Since the early 1990s there have been a number of efforts to define standard approaches for describing and analyzing system architectures. Many of the recent Enterprise Architecture frameworks have some kind of set of views defined, but these sets are not always called "view models".

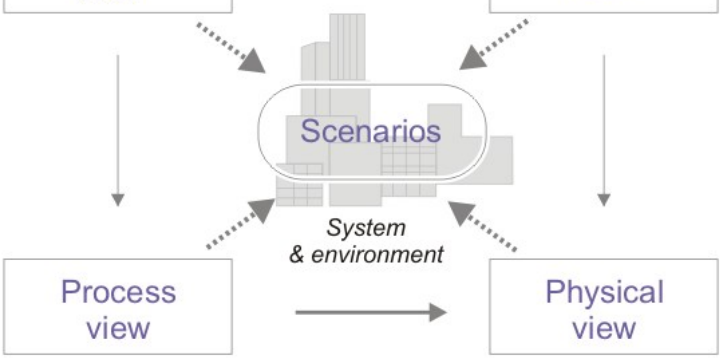


Illustration of the 4+1 view model of architecture.

Standardisation

A first important standard in the field of software architecture and system architecture is IEEE 1471, an IEEE Standard for describing the *architecture of a software-intensive system* approved in 2000. This standard has been adopted by the International Organization for Standardization (ISO) and published as ISO/IEC 42010:2007, still identical to the IEEE 1471:2000. In 2006 a technical committee of the ISO launched a revision of this standard, now published as ISO/IEC/IEEE 42010:2011.

ISO/IEC/IEEE 42010 establishes common terminology for architecture framework and specifies requirements for standardization of frameworks. An architecture framework is defined as:

> *conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders*

An architecture framework is specified by:

1. the relevant stakeholders in the domain,
2. the types of concerns arising in that domain,
3. architecture viewpoints framing those concerns and
4. correspondence rules integrating those viewpoints cited before.

Frameworks conforming to the standard can include methods, tools, definitions, methods and other practices beyond those specified.

Types of enterprise architecture framework

Consortia-developed frameworks

- EABOK (The Guide to the Enterprise Architecture Body of Knowledge) – a U.S. Federal-funded guide to EA in the context of legislative and strategic business requirements.
- Generalised Enterprise Reference Architecture and Methodology (GERAM)
- IDEAS Group – a four-nation effort to develop a common ontology for architecture interoperability
- RM-ODP – the Reference Model of Open Distributed Processing (ITU-T Rec. X.901-X.904 | ISO/IEC 10746) defines an enterprise architecture framework for structuring the specifications of open distributed systems.
- TOGAF – The Open Group Architecture Framework – a widely used framework including an architectural Development Method and standards for describing various types of architecture.
- Good enough architecture methodology – a methodology based on experiences, results and best-practices gathered through real-life implementations of various building blocks that altogether provide a realizable architecture and working solutions.
- ARCON – A Reference Architecture for Collaborative Networks – not focused on a single enterprise but rather on networks of enterprises[10][11]

Open-source frameworks

Enterprise architecture frameworks that are released as open source:

- TRAK – a general systems-oriented framework based on MODAF 1.2 and released under GPL/GFDL.
- MEGAF [12] is an infrastructure for realizing architecture frameworks that conform to the definition of architecture framework provided in ISO/IEC/IEEE 42010.
- Praxeme, an open enterprise methodology, contains an enterprise architecture framework called the Enterprise System Topology (EST)
- GOD, a generalist observation methodology, contains an enterprise architecture framework based on observation, an innovative certified approach provided in the SDFL Department of DUJ.
- SABSA [13] is an open framework and methodology for Enterprise Security Architecture and Service Management, that is risk based and focuses on integrating security into business and IT management.

Proprietary frameworks

- *Avancier Methods (AM)* [14] Processes and documentation advice for enterprise and solution architects, supported by training and certification.
- *Solution Architecting Mechanism (SAM)* [15] – A coherent architecture framework consisting of a set of integral modules.[16]
- Integrated Architecture Framework (IAF) – from Capgemini company in 1993
- CLEAR Framework for Enterprise Architecture – Atos Origin's Enterprise Architecture Framework
- OBASHI – the OBASHI Business & IT methodology and framework
- Information FrameWork (IFW) – conceived by Roger Evernden in 1996
- SAP Enterprise Architecture Framework
- Zachman Framework – an architecture framework, based on the work of John Zachman at IBM in the 1980s

Defense industry frameworks

- DoDAF – the US Department of Defense Architecture Framework
- MODAF – the UK Ministry of Defence Architecture Framework
- NAF – the NATO Architecture Framework
- AGATE – the France DGA Architecture Framework
- DNDAF [17] – the DND/CF Architecture Framework (CAN)

Government frameworks

- Government Enterprise Architecture (GEA) – a common framework legislated for use by departments of the Queensland Government
- FDIC Enterprise Architecture Framework
- Federal Enterprise Architecture Framework (FEAF) – a framework produced in 1999 by the US Federal CIO Council for use within the US Government, not to be confused with the 2002 Federal Enterprise Architecture (FEA) guidance on categorizing and grouping IT investments, issued by the US Federal Office of Management and Budget
- NIST Enterprise Architecture Model
- Treasury Enterprise Architecture Framework (TEAF) – a framework for treasury, published by the US Department of the Treasury in July 2000.[18]
- Nederlandse Overheid Referentie Architectuur (NORA) – a reference framework from the Dutch Government E-overheid NORA [19]

References

[1] The Chief Information Officers Council (1999). Federal Enterprise Architecture Framework Version 1.1 (http://www.cio.gov/documents/fedarch1.pdf). September 1999.

[2] Stephen Marley (2003). Architectural Framework (http://aiwg.gsfc.nasa.gov/esappdocs/RPC/RPC_Workshop_Architecture_Framework.ppt). NASA /SCI. Retrieved 10 Dec 2008.

[3] Patricia A. Oberndorf and Anthony Earl (1998). Department of Veterans Affairs Reference Models (http://www.va.gov/oirm/architecture/ita/seireport.doc). SEI Carnegie Mellon University.

[4] Van Haren (2007) *TOGAF 2007 Edition*. The Open Group. p.142.

[5] Guide to the POSIX Open System Environment (OSE) (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=23985). General info. Accessed 12 Dec 2008.

[6] Rob Thomas and Phil Cullen (2001). " Building an Enterprise Architecture framework (http://cbp.gov/xp/CustomsToday/2001/April/custoday_oit.xml)". In: *US Customs Today* April 2001.

[7] FEA Consolidated Reference Model Document (http://georgewbush-whitehouse.archives.gov/omb/egov/documents/CRM.PDF). whitehouse.gov May 2005.

[8] Péter Bernus (2005). *Knowledge Sharing in the Integrated Enterprise*. p.133-139.

[9] Niles E Hewlett (2006) , The USDA Enterprise Architecture Program (http://www.ocio.usda.gov/p_mgnt/doc/PM_Class_EA_NEH_012506_Final.ppt). PMP CEA, Enterprise Architecture Team, USDA-OCIO. January 25, 2006.

[10] L.M. Camarinha-Matos, H. Afsarmanesh, Collaborative Networks: Reference Modeling, Springer, 2008.

[11] L.M. Camarinha-Matos, H. Afsarmanesh, On reference models for collaborative networked organizations, International Journal Production Research, Vol 46, Nº 9, May 2008, pp 2453–2469.

[12] http://megaf.di.univaq.it/

[13] http://www.sabsa-institute.org/

[14] http://avancier.co.uk

[15] http://doi.ieeecomputersociety.org/10.1109/EDOC.2006.54

[16] Tony Shan and Winnie Hua (2006). *Solution Architecting Mechanism* (http://doi.ieeecomputersociety.org/10.1109/EDOC.2006.54). Proceedings of the 10th IEEE International EDOC Enterprise Computing Conference (EDOC 2006), October 2006, p23-32.

[17] http://www.img-ggi.forces.gc.ca/pub/af-ca/index-eng.asp

[18] US Department of the Treasury Chief Information Officer Council (2000). Treasury Enterprise Architecture Framework (http://www.eaframeworks.com/TEAF/teaf.doc). Version 1, July 2000.

[19] http://www.e-overheid.nl/atlas/referentiearchitectuur/nora/nora.html

Architecture Patterns (EA Reference Architecture)

An **architectural pattern** is a standard design in the field of software architecture. The concept of an architectural pattern has a broader scope than the concept of design pattern. The architectural patterns address various issues in software engineering, such as computer hardware performance limitations, high availability and minimization of a business risk. Some architectural patterns have been implemented within software frameworks.

Definition

Even though an architectural pattern conveys an image of a system, it is not an architecture. An architectural pattern is a concept that solves and delineates some essential cohesive elements of a software architecture. Countless different architectures may implement the same pattern and share the related characteristics. Patterns are often defined as "strictly described and commonly available".[1][2] For example, the layered architecture is a call-and-return style because it defines an overall style to interact. When it is strictly described and commonly available, it is a pattern.

Examples

- Enterprise Architecture framework Patterns in Enterprise Architecture Domains

Here is a list of Architecture Patterns, Design Patterns, and Solution Patterns in the Application and Information Architecture Domains.

Sub-Domain Area	Architecture Pattern Name	Design Patterns	Solution Patterns	Related Patterns
Data Integration/SOA	• ETL (Data Extraction Transformation & Loading)	• Change Data Capture • Near Real-Time ETL • Batch ETL • Data Discovery	• Error handling • Job scheduling • Data validation • Slowly Changing Dimensions Load	• EAI • Master Data Hub • Operational Data Store (ODS) [3] • Data Mart [4] • Data Warehouse
	• MFT			
	• EAI/ESB	• Publish/subscribe • Request/reply • Message Exchange Patterns	• One-Way • Synchronous Request/Response • Basic Callback • Claim Check	• SOA
Data Architecture	• Transaction Data Stores (TDS/OLTP) • Master Data Store • Operational Data Store • Data Mart • Data Warehouse	• Custom Applications Databases • Packaged Application Databases		• ETL • EAI • SOA

Business Intelligence	• Transactional Reporting • Operational Reporting • Analytical Reporting	• Transactional Reporting Data Access • Operational Reporting Data Access • Analytical Reporting Data Access • Analytical Dashboard Data Access • Operational Dashboard Data Access • Data Mining	• Real-Time Dashboards • In-Memory Analytics • Statistical Analysis • Predictive analytics	• ETL • EAI • TDS • Operational Data Store • Data Mart
Master data management	• Master Data Hub	• Master Data Replication • Master Data Services • Master Data Synchronization		• Change Data Capture • EAI • SOA
Data Modeling	• Dimensional Data Modeling • E-R Data Modeling	• Modeling Standards • Naming Conventions		

Some additional examples of architectural patterns:

- Data Mart [4]
- ETL (Data Extraction, Transformation, & Loading) [5]
- Blackboard system
- Event-driven architecture
- Implicit invocation
- Layers
- Model-View-Controller, Presentation-abstraction-control, Model View Presenter, and Model View ViewModel
- Multitier architecture (often three-tier or n-tier)
- Naked objects
- Operational Data Store (ODS) [3]
- Peer-to-peer
- Pipe and filter architecture
- Service-oriented architecture

References

[1] Chang, Chih-Hung; Lu, Chih-Wei; Lin, Chih-Hao; Yang, Ming-Feng; Tsai, Ching-Fu (2008-06). "An Experience of Applying Pattern-based Software Framework to Improve the Quality of Software Development: 4. The Design and Implementation of OS2F" (http://jses.seat.org.tw/index.php/jses/article/viewFile/41/30). *Journal of Software Engineering Studies, Vol. 2, No. 6*. the Third Taiwan Conference on Software Engineering (TCSE07). pp. 185-194. . Retrieved 2012-05-16. "Furthermore, patterns are often defined as something "strictly described and commonly available". For example, layered architecture is a call-and-return style, when it defines an overall style to interact."

[2] "Architectural Patterns: Definition" (http://aahninfotech.com/arct_pattern.html). AAHN INFOTECH (INDIA) PVT. LTD.. . Retrieved 2012-05-16. "Even though an architectural pattern conveys an image of a system, it is not an architecture as such. An architectural pattern is rather a concept that solves and delineates some essential cohesive elements of a software architecture. Countless different architectures may implement the same pattern and thereby share the related characteristics. Furthermore, patterns are often defined as something "strictly described and commonly available"."

[3] http://commons.wikimedia.org/wiki/File:Operational_Data_Store_Architecture_Pattern.jpg

[4] http://commons.wikimedia.org/wiki/File:Datamart_Architecture_Pattern.jpg#file

[5] http://commons.wikimedia.org/wiki/File:ETL_Architecture_Pattern.jpg

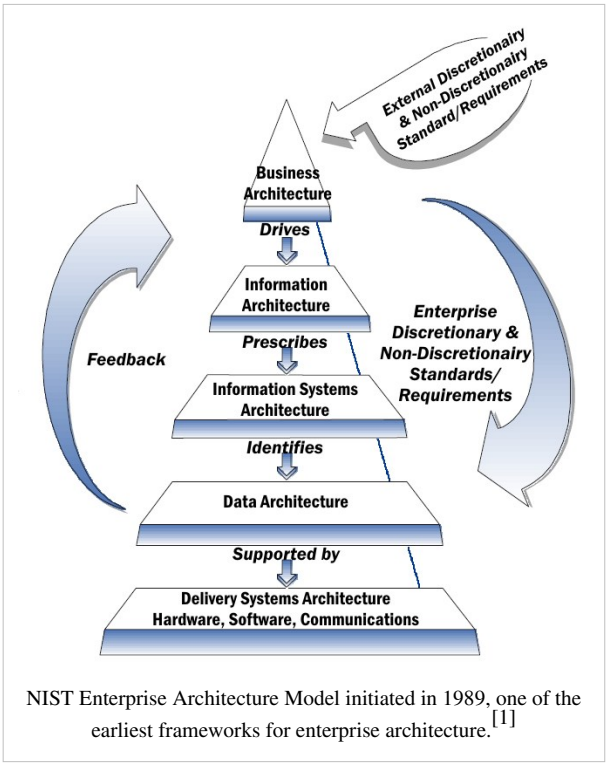
Avgeriou, Paris; Uwe Zdun (2005). "Architectural patterns revisited:a pattern language". *10th European Conference on Pattern Languages of Programs (EuroPlop 2005), Irsee, Germany, July*. Buschmann F., Meunier R., Rohnert H. & Sommerlad P. & Stal M. (1996). *Pattern-Oriented Software Architecture: A System of Patterns* (http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471958697.html). John Wiley & Sons. Bass L., Clements P.,

Kazman R. (2005). *Software Architecture in Practice: Second Edition*. Addison-Wesley.

Frameworks

Enterprise Architecture framework

An **enterprise architecture framework** (EA framework) is a architecture framework which defines how to organize the structure and views associated with an enterprise architecture.



NIST Enterprise Architecture Model initiated in 1989, one of the earliest frameworks for enterprise architecture.[1]

Overview

The three components of the enterprise architecture framework are:[2]

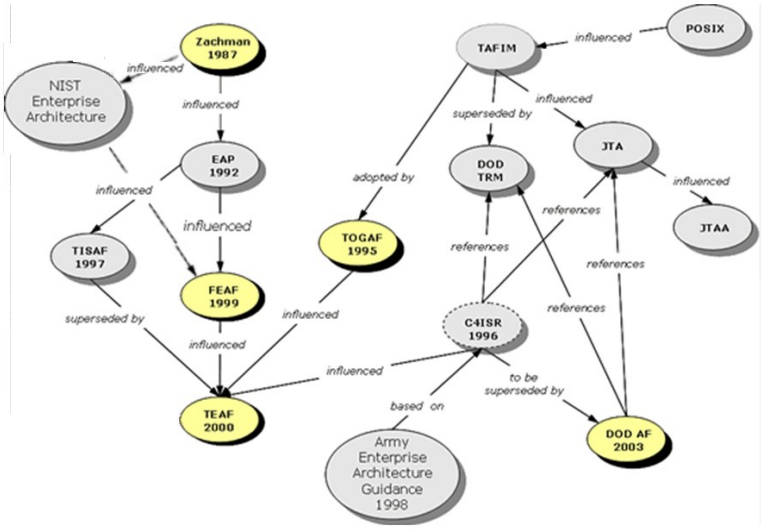
- Views : provide the mechanisms for communicating information about the relationships that are important in the architecture
- Methods : provide the discipline to gather and organize the data and construct the views in a way that helps ensure integrity, accuracy and completeness
- Training/Experience : support the application of method and use of tools

Because the discipline of enterprise engineering and enterprise architecture is so broad, and because enterprises can be large and complex, the models associated with the discipline also tend to be large and complex. To manage this scale and complexity, an architecture framework provides tools and methods that can bring the task into focus and allow valuable artifacts to be produced when they are most needed.

Architecture frameworks are commonly used in IT and information system governance. An organization may wish to mandate that certain models be produced before a system design can be approved. Similarly, they may wish to specify certain views be used in the documentation of procured systems – the U.S. Department of Defense stipulates that specific DoDAF views be provided by equipment suppliers for capital project above a certain value. This discussion is focused for theorical teams who will develop tools and empirical methods of quality achievements.

History

Enterprise architecture started with the Zachman Framework in 1987. Another early implementation of an Enterprise architecture framework was the "Technical Architecture Framework for Information Management" (TAFIM). The first draft of TAFIM was completed in 1991 with the TAFIM Technical Reference Model (TAFIM TRM). This technical reference model wanted to use open systems and new technologies available in the commercial market, to develop a DoD-wide application.[3] The TOGAF TRM was originally derived from the Technical Architecture Framework for Information Management (TAFIM), which in turn was derived from the IEEE model 1003.0[4] or POSIX Open System Environment: a standard "to construct an information processing system, including consumers, system integrators, application developers, system providers, and procurement agencies".[5]



Impression of Enterprise Architecture Frameworks evolution (1987–2003).[2] On the left: The Zachman Framework 1987, NIST Enterprise Architecture 1989, EAP 1992, TISAF 1997, FEAF 1999 and TEAF 2000. On the right: POSIX, TAFIM, JTA, JTAA, TOGAF 1995, DoD TRM and C4ISR 1996, and DoDAF 2003.

In recent years, it has become apparent that a key benefit to be gained from enterprise architecture is the ability to support decision making in changing businesses. Because enterprise architecture brings together business models (e.g. process models, organizational charts, etc.) and technical models (e.g. systems architectures, data models, state diagrams, etc.) it is possible to trace the impact of organizational change on the systems, and also the business impact of changes to the systems.

As this benefit has emerged, many frameworks such as DoDAF, MODAF, or AGATE have adopted a standard meta model which defines the critical architectural elements and the dependencies between them. Applications based on these models can then query the underlying architectural information, providing a simple and strong mechanism for tracing strategies to organizational and technological impacts.

EA framework topics

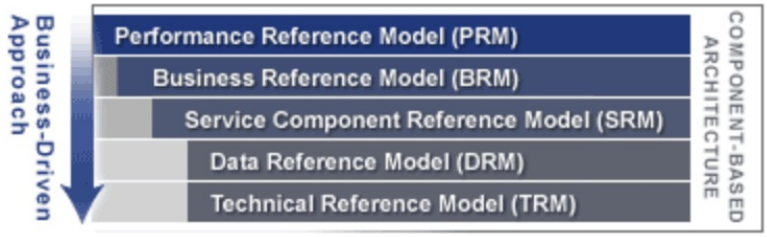
Framework of building codes

Artist impression.[6]

Persons who have ever remodeled their home, know how important building codes, blueprints, and city or county inspections are to successfully complete the project. The architect operates within a "framework" of building codes, preparing blueprints for each phase of the project, from the structural changes to the size and layout of the rooms. Detailed drawings specify plumbing, electrical, and building construction information for the entire structure. Enterprise Architecture works in a similar manner.[6]

An architecture framework for IT affects every aspect of the enterprise. An enterprise architecture framework is similar to building codes that ensure the building is soundly constructed. The IT governance bodies and procedures serve as the city and county inspectors for building improvement projects. Frameworks contain models and standards that will be used to develop IT architecture descriptions. The architecture description is the blueprint.[6]

Architecture domain



Example of the federal enterprise architecture, which has defined five architectural layers.[7]

In the context of the creation of Enterprise Architecture it is common, according to Péter Bernus (2005),[8] to recognise three or four types of architecture, each corresponding to its particular architecture domain. Examples of such domains are:

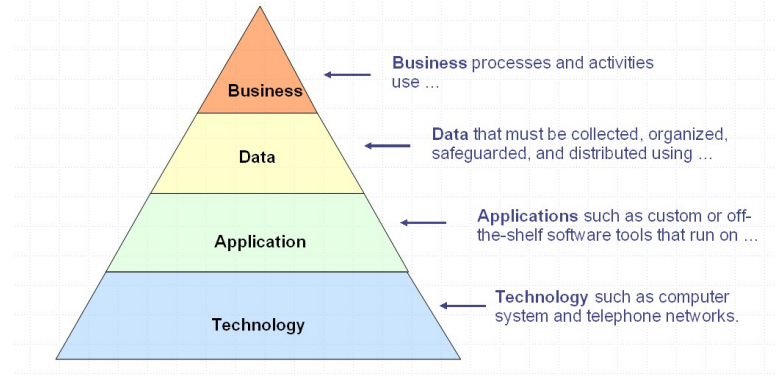
- Business architecture,
- Information systems architecture, often subdivided into
  - Data architecture, and
  - Application architecture,
- and Technical architecture.

Architectural domains are a structuring criterion for a collection of architecture products. They should not be confused with the application domain of the framework as such.[8]

Layers of the enterprise architecture

Contemporary federal guidance suggests thinking about "layers" of the enterprise architecture:[9]

- Business processes and activities
- Data that must be collected, organized, safeguarded, and distributed
- Applications such as custom or off-the-shelf software tools
- Technology such as computer systems and telephone networks

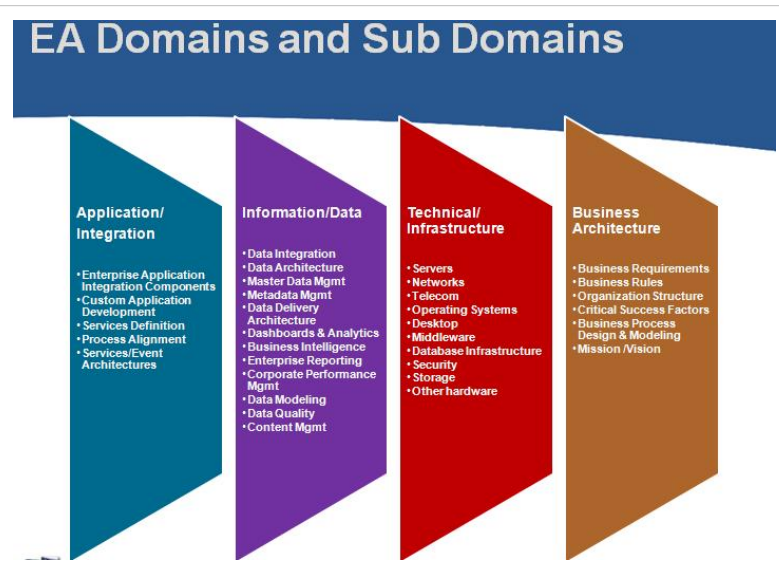


Layers of the enterprise architecture.[9]

The Architecture Domains follow a pattern of decomposition as one goes from top to the bottom of the framework. The ownership can be divided into 4 broad categories: planner's view, owner's view, designer's view and developer's view in this order. All the views are mostly hierarchical in nature. For business view the planner and owner's level is typically called the value chains (which are descriptive by nature). The designer's view of business is also known as the analytical view and there are various standards for modeling this view. One commonly used modeling standard is the Business Process Modeling Notation (BPMN). The designer's view typically represents the execution level which uses standards like Business Process Execution Language (BPEL).

Enterprise architecture domains and subdomains

The application and technology domains (which are not to be confused with business domains) are characterized by domain capabilities and domain services. The capabilities are supported by the services. The application services are also referred to in service-oriented architecture (SOA). The technical services are typically supported by software products.



Enterprise architecture reference architecture with sub domains

The data view starts with the data classes which can be decomposed into data subjects which can be further decomposed into data entities. The basic data model type which is most commonly used is called merda (master entity relationship diagrams assessment, see entity-relationship model). The Class, subject and entity forms a hierarchical view of data. Enterprises may have millions of instances of data entities.

The Enterprise Architecture Reference Traditional Model offers clear distinction between the architecture domains (business, information/data, application/integration and technical/infrastructure). These domains can be further divided into Sub domain disciplines. An example of the EA domain and sub domains is in the image on the right.

Many enterprise architecture teams consist of Individuals with Skills aligned with the Enterprise Architecture Domains and sub-domain disciplines. Here are some examples: enterprise business architect, enterprise

documentational architect, enterprise application architect, enterprise infrastructure architect, enterprise information architect, etc.

An example of the list of reference architecture patterns in the application and information architecture domains are available at Architectural pattern (computer science).

View model

A view model is a framework, which defines the set of views or approaches to be used in systems analysis or systems design or the construction of an enterprise architecture.

Since the early 1990s there have been a number of efforts to define standard approaches for describing and analyzing system architectures. Many of the recent Enterprise Architecture frameworks have some kind of set of views defined, but these sets are not always called "view models".

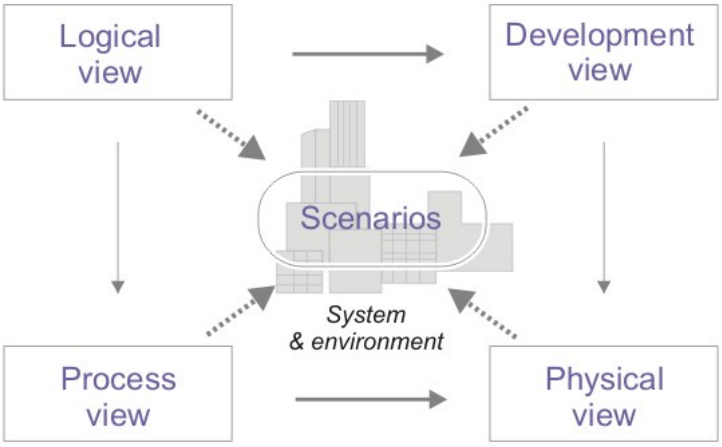


Illustration of the 4+1 view model of architecture.

Standardisation

A first important standard in the field of software architecture and system architecture is IEEE 1471, an IEEE Standard for describing the *architecture of a software-intensive system* approved in 2000. This standard has been adopted by the International Organization for Standardization (ISO) and published as ISO/IEC 42010:2007, still identical to the IEEE 1471:2000. In 2006 a technical committee of the ISO launched a revision of this standard, now published as ISO/IEC/IEEE 42010:2011.

ISO/IEC/IEEE 42010 establishes common terminology for architecture framework and specifies requirements for standardization of frameworks. An architecture framework is defined as:

> *conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders*

An architecture framework is specified by:

1. the relevant stakeholders in the domain,
2. the types of concerns arising in that domain,
3. architecture viewpoints framing those concerns and
4. correspondence rules integrating those viewpoints cited before.

Frameworks conforming to the standard can include methods, tools, definitions, methods and other practices beyond those specified.

Types of enterprise architecture framework

Consortia-developed frameworks

- EABOK (The Guide to the Enterprise Architecture Body of Knowledge) – a U.S. Federal-funded guide to EA in the context of legislative and strategic business requirements.
- Generalised Enterprise Reference Architecture and Methodology (GERAM)
- IDEAS Group – a four-nation effort to develop a common ontology for architecture interoperability
- RM-ODP – the Reference Model of Open Distributed Processing (ITU-T Rec. X.901-X.904 | ISO/IEC 10746) defines an enterprise architecture framework for structuring the specifications of open distributed systems.
- TOGAF – The Open Group Architecture Framework – a widely used framework including an architectural Development Method and standards for describing various types of architecture.
- Good enough architecture methodology – a methodology based on experiences, results and best-practices gathered through real-life implementations of various building blocks that altogether provide a realizable architecture and working solutions.
- ARCON – A Reference Architecture for Collaborative Networks – not focused on a single enterprise but rather on networks of enterprises[10][11]

Open-source frameworks

Enterprise architecture frameworks that are released as open source:

- TRAK – a general systems-oriented framework based on MODAF 1.2 and released under GPL/GFDL.
- MEGAF [12] is an infrastructure for realizing architecture frameworks that conform to the definition of architecture framework provided in ISO/IEC/IEEE 42010.
- Praxeme, an open enterprise methodology, contains an enterprise architecture framework called the Enterprise System Topology (EST)
- GOD, a generalist observation methodology, contains an enterprise architecture framework based on observation, an innovative certified approach provided in the SDFL Department of DUJ.
- SABSA [13] is an open framework and methodology for Enterprise Security Architecture and Service Management, that is risk based and focuses on integrating security into business and IT management.

Proprietary frameworks

- *Avancier Methods (AM)* [14] Processes and documentation advice for enterprise and solution architects, supported by training and certification.
- *Solution Architecting Mechanism (SAM)* [15] – A coherent architecture framework consisting of a set of integral modules.[12]
- Integrated Architecture Framework (IAF) – from Capgemini company in 1993
- CLEAR Framework for Enterprise Architecture – Atos Origin's Enterprise Architecture Framework
- OBASHI – the OBASHI Business & IT methodology and framework
- Information FrameWork (IFW) – conceived by Roger Evernden in 1996
- SAP Enterprise Architecture Framework
- Zachman Framework – an architecture framework, based on the work of John Zachman at IBM in the 1980s

Defense industry frameworks

- DoDAF – the US Department of Defense Architecture Framework
- MODAF – the UK Ministry of Defence Architecture Framework
- NAF – the NATO Architecture Framework
- AGATE – the France DGA Architecture Framework
- DNDAF [17] – the DND/CF Architecture Framework (CAN)

Government frameworks

- Government Enterprise Architecture (GEA) – a common framework legislated for use by departments of the Queensland Government
- FDIC Enterprise Architecture Framework
- Federal Enterprise Architecture Framework (FEAF) – a framework produced in 1999 by the US Federal CIO Council for use within the US Government, not to be confused with the 2002 Federal Enterprise Architecture (FEA) guidance on categorizing and grouping IT investments, issued by the US Federal Office of Management and Budget
- NIST Enterprise Architecture Model
- Treasury Enterprise Architecture Framework (TEAF) – a framework for treasury, published by the US Department of the Treasury in July 2000.[13]
- Nederlandse Overheid Referentie Architectuur (NORA) – a reference framework from the Dutch Government E-overheid NORA [19]

References

[1] The Chief Information Officers Council (1999). Federal Enterprise Architecture Framework Version 1.1 (http://www.cio.gov/documents/fedarch1.pdf). September 1999.

[2] Stephen Marley (2003). Architectural Framework (http://aiwg.gsfc.nasa.gov/esappdocs/RPC/RPC_Workshop_Architecture_Framework.ppt). NASA /SCI. Retrieved 10 Dec 2008.

[3] Patricia A. Oberndorf and Anthony Earl (1998). Department of Veterans Affairs Reference Models (http://www.va.gov/oirm/architecture/ita/seireport.doc). SEI Carnegie Mellon University.

[4] Van Haren (2007) *TOGAF 2007 Edition*. The Open Group. p.142.

[5] Guide to the POSIX Open System Environment (OSE) (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=23985). General info. Accessed 12 Dec 2008.

[6] Rob Thomas and Phil Cullen (2001). " Building an Enterprise Architecture framework (http://cbp.gov/xp/CustomsToday/2001/April/custoday_oit.xml)". In: *US Customs Today* April 2001.

[7] FEA Consolidated Reference Model Document (http://georgewbush-whitehouse.archives.gov/omb/egov/documents/CRM.PDF). whitehouse.gov May 2005.

[8] Péter Bernus (2005). *Knowledge Sharing in the Integrated Enterprise*. p.133-139.

[9] Niles E Hewlett (2006) , The USDA Enterprise Architecture Program (http://www.ocio.usda.gov/p_mgnt/doc/PM_Class_EA_NEH_012506_Final.ppt). PMP CEA, Enterprise Architecture Team, USDA-OCIO. January 25, 2006.

[10] L.M. Camarinha-Matos, H. Afsarmanesh, Collaborative Networks: Reference Modeling, Springer, 2008.

[11] L.M. Camarinha-Matos, H. Afsarmanesh, On reference models for collaborative networked organizations, International Journal Production Research, Vol 46, Nº 9, May 2008, pp 2453–2469.

[12] Tony Shan and Winnie Hua (2006). *Solution Architecting Mechanism* (http://doi.ieeecomputersociety.org/10.1109/EDOC.2006.54). Proceedings of the 10th IEEE International EDOC Enterprise Computing Conference (EDOC 2006), October 2006, p23-32.

[13] US Department of the Treasury Chief Information Officer Council (2000). Treasury Enterprise Architecture Framework (http://www.eaframeworks.com/TEAF/teaf.doc). Version 1, July 2000.

Open Source or Consortia-developed frameworks

Enterprise Architecture Body of Knowledge

The ***Enterprise Architecture Body of Knowledge*** (**EABOK**) is a guide to Enterprise Architecture produced by MITRE (specifically MITRE's Center for Innovative Computing and Informatics), and is substantially funded by US government agencies. It provides a critical review of enterprise architecture issues in the context of the needs of an organization. Because it provides a "big picture" view of needs and methods, some enterprise architecture practitioners recommend it as the "first port of call" for a business establishing an enterprise architecture unit.

Overview

The following paragraph is taken from MITRE's web pages:

> *The Guide to the Enterprise Architecture Body of Knowledge (EABOK) organizes and characterizes the knowledge content of the Enterprise Architecture (EA) discipline. This organization and characterization promotes a consistent view of EA, establishes the scope and bounds of the EA discipline, and places the discipline in the context of related disciplines. The EABOK subdivides EA into knowledge areas and topics within each area, presents an overview of the topic, and provides the reader references for further information. The EABOK is a guide to EA, not the body of knowledge itself.*

The current printable version is dated 06-Feb-2004 and edited by Dr Paula J Hagan. It has been approved for public release; distribution unlimited. New sections and expansions of current "stubs" are under development within MITRE and will be released to the web on an ad-hoc basis.

Alternative titles

While the title of the main printed document is "EABOK", it is also known as:

- "The Guide to the Enterprise Architecture Body of Knowledge" (on the MITRE web pages)
- "The Guide to the (Evolving) Enterprise Architecture Body of Knowledge" (a smaller title on the cover page of the printable document)

Perspective

The key to the EABOK is that it (and the discipline it describes) is evolving (with some of its knowledge areas yet to be fleshed out[1]), and the way it places enterprise architecture *in context*. Because there are so many different frameworks and viewpoints about enterprise architecture, it provides a critique of alternatives (such as between the original Zachman Framework, TOGAF and DODAF). The bibliographies are particularly useful.

It treats Enterprise Architecture as not including merely diagrams and technical descriptions, but gives a holistic view that includes US legislative requirements and guidance, as well as giving technologists a better understanding of business needs with a quick explanation of the Value chain for a business as outlined by Michael Porter.

It is worth reading between the lines of many sections, such as the possible implied criticisms of enterprise architecture units that live solely by the diagram from Zachman's original concepts that enterprise architecture frameworks were necessary:

> *Today Zachman sees his framework as a thinking tool...The Zachman EA Framework has contributed to the organization of several later frameworks and much architectural thinking.*

Another example of possible implied criticism of some EA practitioners:

> *Many* novice *EA practitioners comment that they find the DODAF too complex for a starting point to build an enterprise architecture.* Other *practitioners find the DODAF a good source of product description information to get them started.*

(Italics are not in the original article)

It is *just* such comments that make many experienced information systems and business professionals appreciate the EABOK: while it reviews a range of approaches, it is not frightened to put a personal point-of-view.

Enterprise architecture practitioners should be aware that the discipline has evolved since the most recent publication: perhaps the most notable being the extensions of DODAF, including MODAF, and the work at the Object Management Group to create a model that satisfies both frameworks.

While many of the references to legislation and guidance are US-centric, the issues and the references are useful to government agencies and businesses across the world.

References

[1] Knowledge Areas (http://www.mitre.org/tech/eabok/knowledge_areas.html), MITRE, USA.

External links

- Guide to the (Evolving) Enterprise Architecture Body of Knowledge (http://www.mitre.org/work/tech_papers/tech_papers_04/04_0104/04_0104.pdf) (Current as of 2007-01-01)
- "The Edge" (http://www.mitre.org/news/the_edge), an EA newsletter from MITRE

Generalised Enterprise Reference Architecture and Methodology

Generalised Enterprise Reference Architecture and Methodology (GERAM) is a generalised Enterprise Architecture framework for enterprise integration and business process engineering. It identifies the set of components recommended for use in enterprise engineering.[1]

This framework is developed in the 1990s by an IFAC/IFIP Task Force on Architectures for Enterprise Integration. The development starting with the evaluation of existing frameworks for enterprise integration which was developed into an overall definition of a socalled "generalised architecture", which was named GERAM for "Generalised Enterprise Reference Architecture and Methodology".[2]

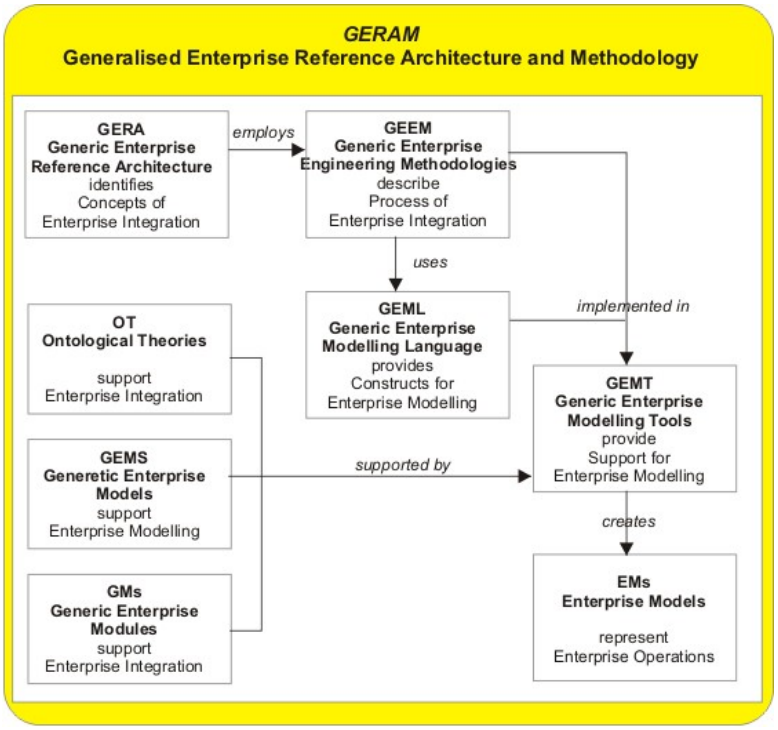


Fig 1. GERAM Framework: This set of components is identified in the first image and briefly described in the following. Starting from defined concepts to be used in enterprise integration (GERA), GERAM distinguishes between the methodologies for enterprise integration (GEEM) and the languages used to describe structure, contents and behaviour of the enterprise (GEML).

Overview

Enterprise modelling is seen as the major item in enterprise engineering and integration. Therefore, both the methodologies and the corresponding languages will be implemented in enterprise modelling tools (GEMT) which will support the enterprise integration process. Ontological theories (OT), generic enterprise models (GEMs) and generic modules (GMs) will support the modelling process by providing means for more efficient modelling.[1]

The modelling process will result in an enterprise model (EM) which represents all or part of the enterprise operation. These models will allow simulation of operational alternatives and thereby their evaluation leading to the optimum structure, contents and behaviour of the enterprise operation. GERAM provides a generic description of all the elements recommended in enterprise engineering and integration.

Generalised Enterprise Reference Architecture and Methodology (GERAM) is an enterprise-reference architecture that models the whole life history of an enterprise integration project from its initial concept in the eyes of the entrepreneurs who initially developed it, through its definition, functional design or specification, detailed design, physical implementation or construction, and finally operation to obsolescence. The architecture aims to be a relatively simple framework upon which all the functions and activities involved in the aforementioned phases of the life of the enterprise-integration project can be mapped. It also will permit the tools used by the investigators or practitioners at each phase to be indicated. The architecture defined will apply to projects, products, and processes; as well as to enterprises.[3]

History

Generalised Enterprise Reference Architecture and Methodology (GERAM) is developed in the 1990s by an IFAC/IFIP Task Force on Architectures for Enterprise Integration with Peter Bernus, James G. Nell and others. The IFAC/IFIP Task Force on Architectures for Enterprise Integration was establishment in 1990 and had studied enterprise-reference architectures ever since.[3]

In its work the Task Force has established the requirements to be satisfied by candidate enterprise-reference architectures and their associated methodologies to fulfill the needs of industry for such aids to enterprise integration. The result has been called GERAM, for "Generalized Enterprise-Reference Architecture and Methodology", by the Task Force. The Task Force has shown that such an architecture is feasible and that several architectures presently available in the literature can already or potentially can fulfil such requirements.[3]

The development of enterprise-reference architecture has evolved from the development of Design Methodology for Advanced Manufacturing Systems in the 1980s,[4] such as CIMOSA, the Open System Architecture for CIM.[5][6] The GERAM framework was first published by Peter Bernus and Laszlo Nemes in 1994.[2]

GERAM Topics

GERAM Components

The Generalised Enterprise Reference Architecture and Methodology (GERAM) consists of a series of eight main components, as shown in figure 1:

- *Generic Enterprise Reference Architecture (GERA)* : Defines the enterprise related generic concepts recommended for use in enterprise integration projects. These concepts include enterprise systems life cycle; business process modeling; modeling languages for different users of the architecture (business users, system designers, IT modeling specialists, others); integrated model representation in different model views.
- *Generic Enterprise Engineering Methodologies (GEEM)* : Describe the generic processes of enterprise integration. These methodologies may be described in terms of process models with detailed instruction for each step of the integration process.
- *Generic Enterprise Modeling Languages (GEML)* : Define the generic constructs (building blocks) for enterprise modeling adapted to the different needs of people creating and using enterprise models.
- *Generic Enterprise Modeling Tools (GEMT)* : Define the generic implementation of enterprise-integration methodologies and modeling languages and other support for creation and use of enterprise models.
- *Enterprise Models (EM)* : Represents the enterprise operation. These models will be represented using generic modeling language constructs.
- *Ontological Theories (OT)* : Formalise the most generic aspects of enterprise-related concepts in terms of essential properties and axioms.
- *Generic Enterprise Models (GEMs)* : Identify reference models (partial models) which capture concepts common to many enterprises. GEMs will be used in enterprise modeling to increase modeling process efficiency.
- *Generic Modules (GMs)* : Identify generally applicable products to be employed in enterprise integration (e.g. tools, integrating infrastructures, others.).

Generic Enterprise Reference Architecture

Generic Enterprise Reference Architecture (GERA) defines the enterprise related generic concepts recommended for use in enterprise integration projects. These concepts include life cycle; enterprise entity types, enterprise modelling with business process modelling; integrated model representation in different model views and modelling languages for different users of the enterprise architecture (business users, system designers, IT modelling specialists, among others).[1]

Life-Cycle Concept

Provides for the identification of the life-cycle phases for any enterprise entity from entity conception to its final end. The Figure 2: GERA Life-Cycle Concept, shows the GERA life cycle phases of enterprise entities. A total of 9 life cycle phases has been defined.

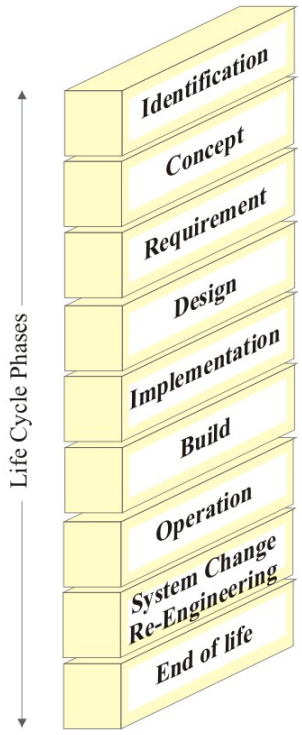


Fig 2. GERA Life-Cycle Concept.

- Identification phase allows the identification of the enterprise business or any part of it in terms of its relation to both its internal and external environment. This includes the definition general commitments of the integration or engineering activities to be carried out in relevant projects.
- Concept phase provides for the presentation of the management visions, missions, values, operational concepts (build/buy, etc.), policies, plus others.
- Requirement phase allows the description of operational processes and collection of all their functional, behavioural, informational and capability requirements.
- Design phase is the specification of operational system with all its components satisfying the above requirements. Process and resources alternatives may be specified which provide operational alternatives to be used during the operation.
- Implementation phase describes the real operational system which may deviate from the designed system due to enterprise preferences or availability of components.
- Build phase supports the system manifestation, physical implementation of resources, testing and validation for the designed processes and the subsequent release for operation.
- Operation phase employs the released operational processes and the provided resources to support the life cycle phases of the enterprise products.
- System Change/Re-Engineering phase allows to modify or re-engineer the operational processes according to newly identified needs or capabilities provided by new technologies.
- End of Life phase supports the recycling or disposal of the operational system at the ending of its use in the enterprise operation. This phase has to provide concepts for recycling and/or disposal of all or part of the system.

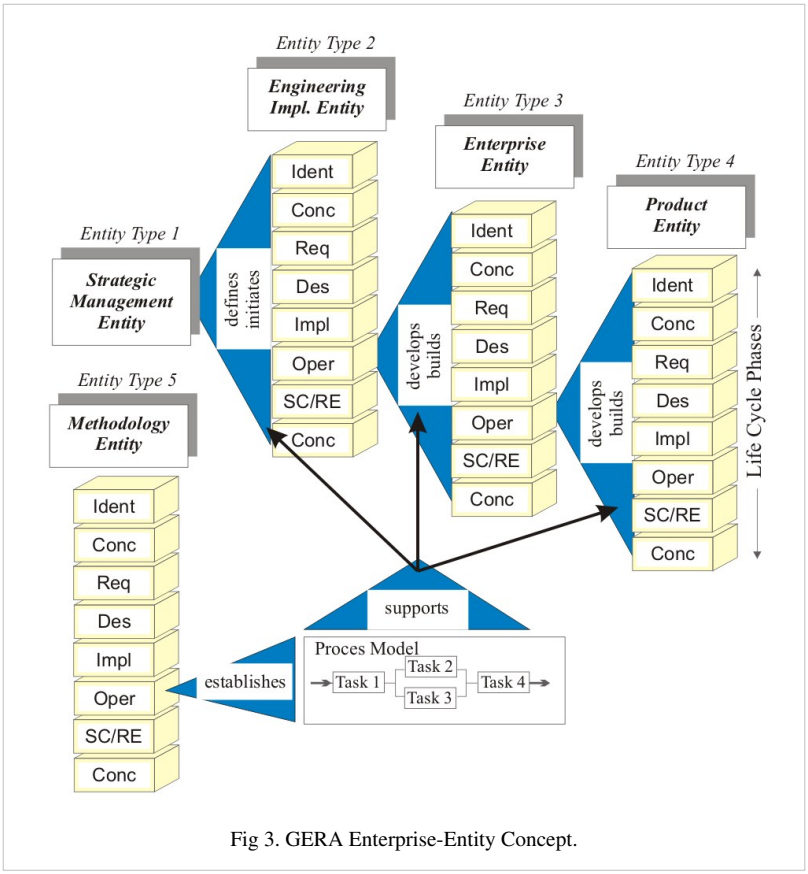


Fig 3. GERA Enterprise-Entity Concept.

Enterprise Entity Type Concept

Identifies entity types to be used in enterprise engineering and enterprise integration. Adopting a recursive view of integration altogether five entity types with their associated life-cycles can be identified. The recursiveness of the first four entity types can be demonstrated by identifying the role of the different entities, their products and the relations between them. Figure 3: GERA Enterprise Entity Concept, shows the GERA life cycle phases of enterprise entities. A total of 9 life cycle phases has been defined.

- Strategic Enterprise Management Entity (type 1): defines the necessity and the starting of any enterprise engineering effort.
- Enterprise Engineering/Integration Entity (type 2): provides the means to carry out the enterprise entity type 1. It employs methodologies (type 5 entity) to define, design, implement and build the operation of the enterprise entity (type 3 entity).
- Enterprise Entity (type 3): is the result of the operation of entity type 2. It uses methodologies (entity type 5) and the operational system provided by entity type 2 to define, design, implement and build the products (services) of the enterprise (type 4 entity).
- Product Entity (type 4): is the result of the operation of entity type 3. It represents all products (services) of the enterprise.
- Methodology Entity (type 5): represents the methodology to be employed in any enterprise entity type.

Figure 3 represents the chain of enterprise entity developments. The type 1 entity will always start creation of any lower level entity by identifying goal, scope and objectives for the particular entity. Development and implementation of a new enterprise entity (or new business unit) will then be done by a type 2 entity; whereas a type 3 entity will be responsible for developing and manufacturing a new product (type 4 entity). With the possible exception of the type 1 entity all enterprise entities will have an associated entity-life cycle. However, it is always the operational phase of the entity-life cycle in which the lower entity is defined, created, developed and built. The operation itself is supported by an associated methodology for enterprise engineering, enterprise operation, product development and production support

Figure 3 also shows the life cycle of the methodology (type 5 entity) and the process model developed during the early life cycle phases of the methodology. However, there must be a clear distinction between the life cycle of the methodology with its different phases and its process model. The latter is used to support the operational phase of a particular enterprise entity. The operational relations of the different entity types are also shown in Figure 4: GERA Enterprise Entity Concept (Type 3), which demonstrates the contributions of the different entities to the type 3 entity life-cycle phases. The manufacturing entity itself produces the enterprise product in the course of its operation phase (type 3 entity).

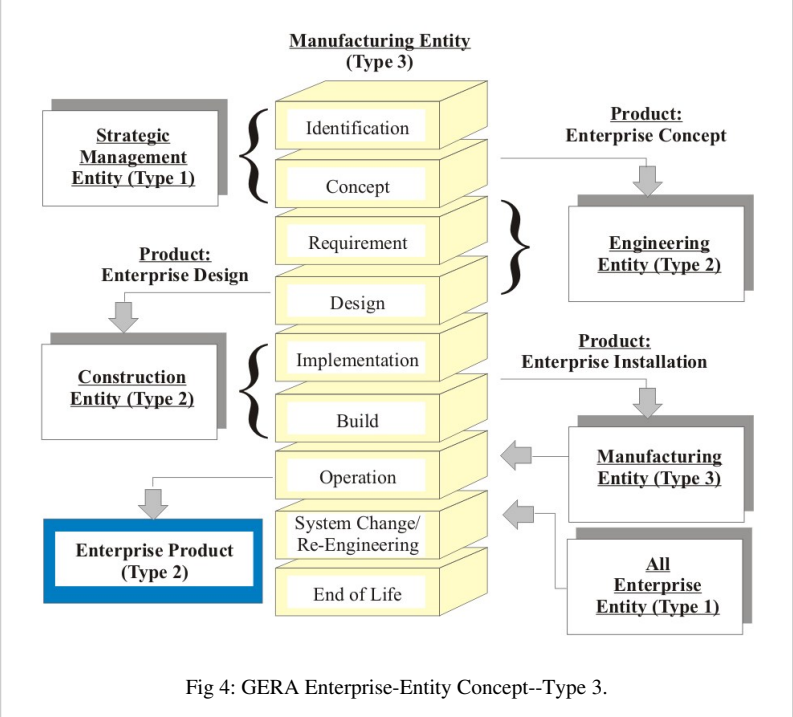


Fig 4: GERA Enterprise-Entity Concept--Type 3.

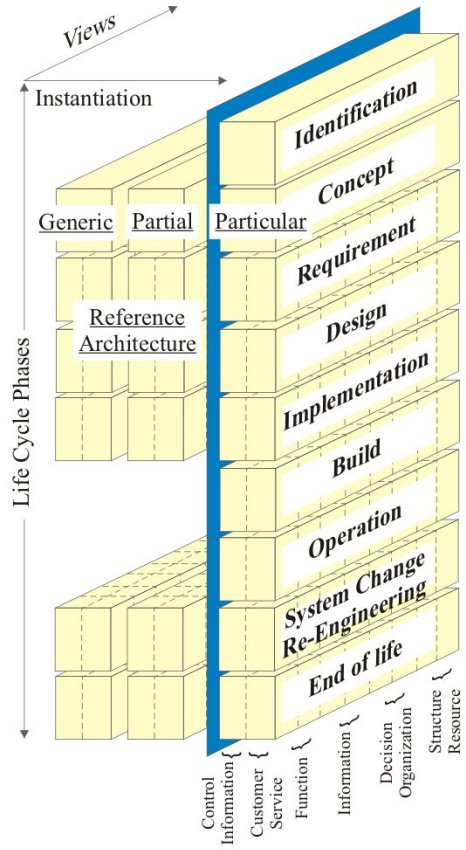


Fig 5. GERA Generic-Reference-Architecture Concept.

Enterprise Modelling concept

Enterprise Modelling concept provides process models of enterprise operations. Process oriented modelling allows to represent the operation of enterprise entities and entity types in all its aspects: functional, behaviour, information, resources and organisation. Models which can be used for decision support by evaluating operational alternatives or for model driven operation control and monitoring.

To hide complexity of the resulting model it will be presented to the user in different sub-sets (views). This view concept is shown in Figure 5 : GERA Generic Reference Architecture Concept. It is applicable during all phases of the life cycle. Please note that the views will be generated from the underlying integrated model and any model manipulation. That means any change being done in one particular view will be reflected in all relevant aspects of the model. The GERA life cycle model has defined four different views: function, information, decision/organisation and resource/structure. Other views may be defined if needed and supported by the modelling tool. In addition, the life cycle model of GERA provides for two different categories of modelling: operation control and customer-service related.

Modelling Language concept

Modelling languages increase the efficiency of enterprise modelling. In addition they allow a common representation of the enterprise operation. Modelling languages have to accommodate different users of enterprise models; for example, business users, system designers, and IT-modelling specialists.

Modelling languages have to support the modelling of all entity types across all phases of their respective life cycles. In addition, modelling languages have to provide generic constructs as well as macro constructs (GEMs) build from generic ones. The latter will further enhance modelling productivity.

Figure 5 shows the reference architecture for those enterprise entity life cycle phases which require generic constructs. The partial level shows the place of the GEMs in the reference architecture. The particular level indicates the life cycle phases of the enterprise entity itself.

Generic Enterprise Engineering Methodologies

Generic Enterprise engineering methodologies (GEEM) describe the process of enterprise integration and, according to the GERAM framework (Figure 1), will result in a model of the enterprise operation. The methodologies will guide the user in the engineering task of enterprise modelling and integration. Different methodologies may exist which will guide the user through the different tasks required in the integration process.[1]

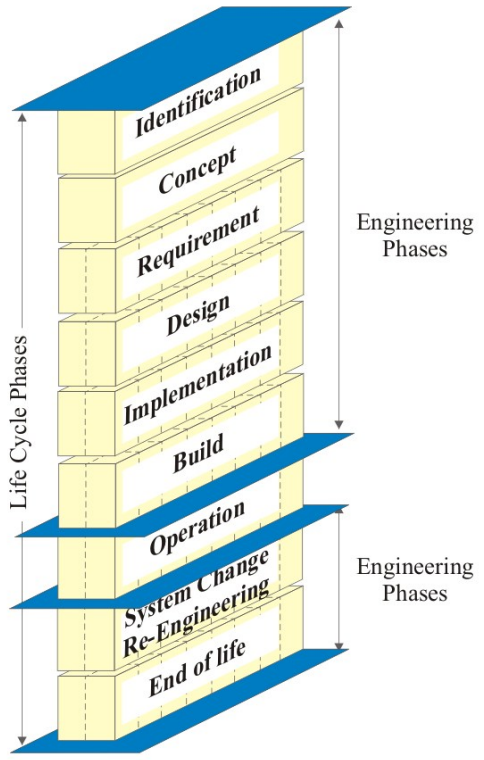


Fig 6. Enterprise Engineering and the Life-Cycle Concept.

Enterprise-engineering methodologies should orient themselves on the life-cycle concept identified in GERA and should support the different life cycle phases shown in Figure 2. The enterprise integration process itself is usually directed towards the enterprise entity type 3 (see above) operation and carried out as an enterprise engineering task by an enterprise entity type 2 (Figures 2 and 4). The integration task may start at any relevant engineering phase (indicated in Figure 6: Enterprise Engineering and the Life-Cycle Concept.) of the entity life cycle and may employ any of those phases. Therefore the processes relating to the different phases of enterprise engineering should be independent of each other to support different sequences of engineering tasks.

Enterprise engineering methodologies may be described in terms of process models with detailed instruction for each step of the integration process. This allows not only a very good representation of the methodology for its understanding, but provides for identification of information to be used and produced, resources needed and relevant responsibilities to be assigned for the integration process. Process representation of methodologies should employ the relevant modelling language discussed below.

Generic Enterprise-Modelling Language

Generic enterprise modelling languages (GEML) define generic constructs (building blocks) for enterprise modelling. Generic constructs which represent the different elements of the operation improve both modelling efficiency and model understanding. These constructs have to be adapted to the different needs of people creating and using enterprise models. Therefore different languages may exist which accommodate different users (e.g. business users, system designers, IT modelling specialists, others).[1]

Modelling the enterprise operation means to describe its processes and the necessary information, resources and organisational aspects. Therefore modelling languages have to provide constructs capable of capturing the semantics of enterprise operations. This is especially important if enterprise models are to support the enterprise operation itself.

Model-based decision support and model-driven operation control and monitoring require modelling constructs which are supporting the end users and which represent the operational processes according to the users perception.

Modelling languages increase the efficiency of enterprise modelling. In addition they allow a common representation of the enterprise operation. Modelling languages have to support the modelling of all entity types across all phases of their respective life cycles. In addition, modelling languages have to provide generic constructs as well as macro constructs (GEMs) build from generic ones. The latter will further enhance modelling productivity.

Generic Enterprise-Modelling Tool

Generic enterprise modelling tools (GEMT) define the generic implementation of the enterprise integration methodologies and modelling languages and other support for creation and use of enterprise models. Modelling tools should provide user guidance for both the modelling process itself and for the operational use of the models. Therefore enterprise modelling tools designs have to encompass not only the modelling methodology, but should provide model enactment capability for simulation of operational processes as well. The latter should also include analysis and evaluation capabilities for the simulation results.[1]

Enterprise Models

Enterprise models (EMs) represent the enterprise operation mostly in the form of business processes. However, in certain cases other representations may be suitable as well. Business processes will be represented using the generic modelling-language constructs defined above for the relevant engineering methodology. Enterprise operations are usually rather complex and therefore difficult to understand if all relevant aspects of the operation are represented in a common model. To reduce the model complexity for the user, different views should be provided which allow the users only to see the aspect of concern.[1]

Ontological Theories

Ontological theories (OT) formalise the most generic aspects of enterprise related concepts in terms of essential properties and axioms. Ontological theories may be considered as 'meta-models' since they consider facts and rules about the facts and rules of the enterprise and its models.[1]

Generic Enterprise Models

Generic enterprise models (GEMs) identify reference models (partial models) which capture concepts common to many enterprises. GEMs will be used in enterprise modelling to increase modelling process efficiency.[1]

Generic Modules

Generic Modules (GMs) identify generally applicable products to be employed in enterprise integration (e.g. tools, integrating infrastructures, others.).[1]

References

[1] J.G. Nell, NIST (1997). " An Overview of GERAM (http://www.mel.nist.gov/workshop/iceimt97/ice-gera.htm)" ICEIMT'97 International Conference on Enterprise Integration Modelling Technology 1997. Updated 30 January 1997

[2] P. Bernus, and L. Nemes (1994). "A Framework to Define a Generic Enterprise Reference Architecture and Methodology". In: *Proceedings of the International Conference on Automation, Robotics and Computer Vision (ICARCV'94)*, Singapore, November 10–12, 1994.

[3] J.G. Nell (2006). " Requirements and Methodology for Enterprise-Reference Architectures: A New Work Item Proposal (http://www.mel.nist.gov/sc5wg1/np_geram.htm)". updated 20 May 1996.

[4] Doumeingts, G., Vallespir, B., Darracar, D., M., "Design Methodology for Advanced Manufacturing Systems", Computers in Industry, Vol. 9, pp. 271-296, December 1987.

[5] AMICE Consortium (1989). *Open System Architecture for CIM, Research Report of ESPRIT Project 688*, Vol. 1, Springer-Verlag.

[6] AMICE Consortium (1991), *Open System Architecture, CIMOSA, AD 1.0, Architecture Description*, ESPRIT Consortium AMICE, Brussels, Belgium.

[7] http://www.nist.gov

Further reading

- F.B. Vernadat (1996). "Enterprise Modeling and Integration: Principles and Applications", Chapman & Hall, London. ISBN 0-412-60550-3
- T.J. Williams and Hong Li, *A Specification and Statement of Requirements for GERAM (The Generalised Enterprise Reference Architecture and Methodology) with all Requirements illustrated by Examples from the Purdue Enterprise Reference Architecture and Methodology PERA*, REPORT NUMBER 159 Purdue Laboratory for Applied Industrial Control November 1995, Version 1.1
- D. Shorter, Editor, "An evaluation of CIM modelling constructs - Evaluation report of constructs for views according to ENV 40 003", In: *Computers in Industry* - Vol. 24, Nrs 2-3
- T.J. Williams, et al., "Architectures for integrating manufacturing activities and enterprises", In: *Computers in Industry* - Vol. 24, Nrs 2-3
- *ENV 40 003 Computer Integrated Manufacturing* - Systems Architecture - Framework for Enterprise Modelling CEN/CENELEC, 1990
- *ENV 12 204 Advanced Manufacturing Technology* - Systems Architecture - Constructs for Enterprise Modelling CEN TC 310/WG1, 1995
- Charles J. Petrie, Jr (1992). *Enterprise Integration Modelling; ICEIMT Conference Proceedings*, The MIT Press. ISBN 0-262-66080-6

External links

- GERAM: Generalised Enterprise Reference Architecture and Methodology Version 1.6.3 (http://www.cit.gu.edu.au/~bernus/taskforce/geram/versions/geram1-6-3/v1.6.3.html). by Peter Bernus, March 1999.

IDEAS Group

The **IDEAS Group** is the International Defence Enterprise Architecture Specification for exchange Group. The deliverable of the project is a data exchange format for military Enterprise Architectures. The scope is four nation (plus NATO as observers) and covers MODAF (UK), DoDAF (USA), DNDAF [17] (Canada) and the Australian Defence Architecture Framework. The initial scope for exchange is the architectural data required to support coalition operations planning -

- Systems - communications systems, networks, software applications, etc.
- Communications links between systems
- Information specifications - the types of information (and their security classifications) that the comms architecture will handle
- Platforms & facilities.
- System & operational functions (activities)
- People & organizations
- Architecture meta-data - who owns it, who was the architect, name, version, description, etc.

The work has begun with the development of a formal ontology to specify the data exchange semantics. The W3C Resource Description Framework (RDF) and Web Ontology Language (OWL) will be the format used for data exchange. A demonstration of multinational interoperability is scheduled for September 2007, based on exchanging process models for casualty tracking.

The Need for Architecture Interoperability

The need for IDEAS was identified in 2005 by the Australian, Canadian, UK & US defence departments. The main purpose of IDEAS is to support coalition military operations planning. The ability to exchange architectures between countries enables better understanding of each others capabilities, communications mechanisms and standard procedures.

Military Application

- **Sharing of standard operating procedures and doctrine.** Each nation has its own operating procedures, which are usually represented in process models (e.g. a DoDAF OV-5 Product). An ability to share these processes between partner nations enables better understanding and more efficient joint operations.
- **Sharing of systems information.** The nations generally use different communications systems, weapon systems and platforms. An ability to share technical information (within classification limits) enables better understanding of how partner nations' systems communicate and function. This allows for better orchestration of sensors and effects across the coalition.
- **Change management.** The procurement cycles in each of the nations are generally not coordinated with each other. Ability to share future systems information amongst partner nations allows for better forward planning.
- **Identification of system options.** In planning a coalition operation, it is useful for the planner to have an accurate picture of each nation's capability contribution. This allows for selection of best capability, and reduction in redundancy and start-up effort.
- **Identification of network configuration.** It is important for each nation to have an understanding of the comms laydown for an operation. An ability to provide each nation with the same understanding of the comms structure means there is less re-work and less opportunity for gaps in understanding between the nations
- **Assessment of Relative Performance.** IDEAS will enable exchange of complete Enterprise Architecture models, potentially allowing simulation of capability prior to operational commitment.

- **Force Structures.** Coalition nations will be able to share organisational structures and orders of battle with partners (again, subject to issues of security classification).

Ontology

IDEAS is a formal, higher-order, 4D (see four dimensionalism) ontology. It is extensional (see Extension (metaphysics)), using physical existence as its criterion for identity. In practical terms, this means the ontology is well suited to managing change over time and identifying elements with a degree of precision that is not possible using names alone.

The ontology is being built using the BORO Method which has proven useful for the multi-disciplinary team working on IDEAS. BORO forces the ontology developer to consider each concept in terms of its physical extent. This means there can be no argument about names or meaning - something either exists or it doesn't. The BORO Method also deals with classes and relationships by tracing them back to their members (classes) or ends (relationships).

Implementation

To date, there have been three IDEAS implementations:

- IDEAS Ontology Development Plug-in for Sparx Enterprise Architect. The Beta test version can be downloaded from http://www.modelfutures.com/software/
- An OV-5 export interface for Telelogic System Architect. The interface was developed by Silver Bullet inc. under contract to the DoD
- An OV-5 import/export interface for Sparx Enterprise Architect developed by Model Futures for MOD (under sub-contract to Serco Consulting)
- The MOD Ontology Demonstrator - used the IDEAS model to demonstrate a simple geopolitical ontology. It can be downloaded from http://www.modaf.com/News/69/mod-ontology-demonstrator-released
- The UK MOD AV-2 demonstrator - implemented a shared ontology based on IDEAS that is used to populate AV-2 in MODAF

IDEAS Publications & Presentations

The IDEAS work has been presented at a number of conferences[1][2]. It has also been cited in a Cutter Consortium white paper[3] and in a book on Systems Engineering from Springer Verlag[4]

Notes

[1] Bailey, Ian; Partridge, Chris. *Working with Extensional Ontology for Defence Applications*. Ontology in Intelligence Conference, 2009, GMU, Fairfax, VA

[2] NATO C3 Agency Workshop on ontology, NC3A The Hague, March 2008 - http://www.ideasgroup.org/file_download/3/MOD+Ontology.pdf

[3] http://www.cutter.com/offers/forensicIS.html

[4] Tolk, Andreas; Jain, Lakhmi C. (Eds.) *Intelligence-Based Systems Engineering*. Springer, 2011, ISBN 978-3-642-17930-3

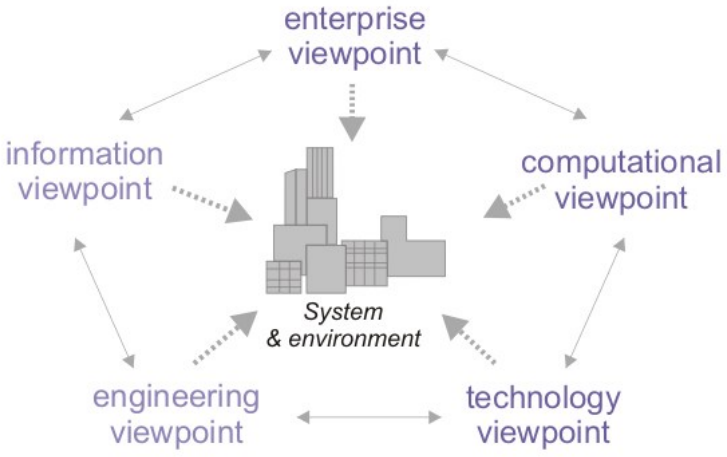
External links

- IDEAS Group (http://www.ideasgroup.org) Official site
- Presentation on IDEAS (http://www.integrated-ea.com/Previous-Years) by Dave McDaniel from Integrated EA 2008
- IDEAS Group on LinkedIn (http://www.linkedin.com/groups?gid=3122051)
- Conceptual Interoperability

RM-ODP

Reference Model of Open Distributed Processing (RM-ODP) is a reference model in computer science, which provides a co-ordinating framework for the standardization of open distributed processing (ODP). It supports distribution, interworking, platform and technology independence, and portability, together with an enterprise architecture framework for the specification of ODP systems.

RM-ODP, also named *ITU-T Rec. X.901-X.904* and *ISO/IEC 10746*, is a joint effort by the International Organization for Standardization (ISO), the International Electrotechnical Commission (IEC) and the Telecommunication Standardization Sector (ITU-T) .[1]



The RM-ODP view model, which provides five generic and complementary viewpoints on the system and its environment.

Overview

The RM-ODP is a reference model based on precise concepts derived from current distributed processing developments and, as far as possible, on the use of formal description techniques for specification of the architecture. Many RM-ODP concepts, possibly under different names, have been around for a long time and have been rigorously described and explained in exact philosophy (for example, in the works of Mario Bunge) and in systems thinking (for example, in the works of Friedrich Hayek). Some of these concepts -- such as abstraction, composition, and emergence -- have recently been provided with a solid mathematical foundation in category theory.

RM-ODP has four fundamental elements:

- an object modelling approach to system specification;
- the specification of a system in terms of separate but interrelated viewpoint specifications;
- the definition of a system infrastructure providing distribution transparencies for system applications; and
- a framework for assessing system conformance.

The RM-ODP family of recommendations and international standards defines a system of interrelated essential concepts necessary to specify open distributed processing systems and provides a well-developed enterprise architecture framework for structuring the specifications for any large-scale systems including software systems.

History

Much of the preparatory work that led into the adoption of RM-ODP as an ISO standard was carried out by the Advanced Networked Systems Architecture (ANSA) project. This ran from 1984 until 1998 under the leadership of Andrew Herbert (now MD of Microsoft Research in Cambridge), and involved a number of major computing and telecommunication companies. Parts 2 and 3 of the RM-ODP were eventually adopted as ISO standards in 1996. Parts 1 and 4 were adopted in 1998.

RM-ODP Topics

RM-ODP standards

RM-ODP consists of four basic ITU-T Recommendations and ISO/IEC International Standards:[2][3][4][5]

1. Overview[6]: Contains a motivational overview of ODP, giving scoping, justification and explanation of key concepts, and an outline of the ODP architecture. It contains explanatory material on how the RM-ODP is to be interpreted and applied by its users, who may include standard writers and architects of ODP systems.
2. Foundations[7]: Contains the definition of the concepts and analytical framework for normalized description of (arbitrary) distributed processing systems. It introduces the principles of conformance to ODP standards and the way in which they are applied. In only 18 pages, this standard sets the basics of the whole model in a clear, precise and concise way.
3. Architecture[8]: Contains the specification of the required characteristics that qualify distributed processing as open. These are the constraints to which ODP standards must conform. This recommendation also defines RM-ODP viewpoints, subdivisions of the specification of a whole system, established to bring together those particular pieces of information relevant to some particular area of concern.
4. Architectural Semantics[9]: Contains a formalization of the ODP modeling concepts by interpreting many concepts in terms of the constructs of the different standardized formal description techniques.

Viewpoints modeling and the RM-ODP framework

Most complex system specifications are so extensive that no single individual can fully comprehend all aspects of the specifications. Furthermore, we all have different interests in a given system and different reasons for examining the system's specifications. A business executive will ask different questions of a system make-up than would a system implementer. The concept of RM-ODP viewpoints framework, therefore, is to provide separate viewpoints into the specification of a given complex system. These viewpoints each satisfy an audience with interest in a particular set of aspects of the system. Associated with each viewpoint is a viewpoint language that optimizes the vocabulary and presentation for the audience of that viewpoint.

Viewpoint modeling has become an effective approach for dealing with the inherent complexity of large distributed systems. Current software architectural practices, as described in IEEE 1471, divide the design activity into several areas of concerns, each one focusing on a specific aspect of the system. Examples include the "4+1" view model, the Zachman Framework, TOGAF, DoDAF and, of course, RM-ODP.

A viewpoint is a subdivision of the specification of a complete system, established to bring together those particular pieces of information relevant to some particular area of concern during the analysis or design of the system. Although separately specified, the viewpoints are not completely independent; key items in each are identified as related to items in the other viewpoints. Moreover, each viewpoint substantially uses the same foundational concepts (defined in Part 2 of RM-ODP). However, the viewpoints are sufficiently independent to simplify reasoning about the complete specification. The mutual consistency among the viewpoints is ensured by the architecture defined by RM-ODP, and the use of a common object model provides the glue that binds them all together.

More specifically, the RM-ODP framework provides five generic and complementary viewpoints on the system and its environment:

- The *enterprise viewpoint*, which focuses on the purpose, scope and policies for the system. It describes the business requirements and how to meet them.
- The *information viewpoint*, which focuses on the semantics of the information and the information processing performed. It describes the information managed by the system and the structure and content type of the supporting data.
- The *computational viewpoint*, which enables distribution through functional decomposition on the system into objects which interact at interfaces. It describes the functionality provided by the system and its functional decomposition.
- The *engineering viewpoint*, which focuses on the mechanisms and functions required to support distributed interactions between objects in the system. It describes the distribution of processing performed by the system to manage the information and provide the functionality.
- The *technology viewpoint*, which focuses on the choice of technology of the system. It describes the technologies chosen to provide the processing, functionality and presentation of information.

RM-ODP and UML

Currently there is growing interest in the use of UML for system modelling. However, there is no widely agreed approach to the structuring of such specifications. This adds to the cost of adopting the use of UML for system specification, hampers communication between system developers and makes it difficult to relate or merge system specifications where there is a need to integrate IT systems.

Although the ODP reference model provides abstract languages for the relevant concepts, it does not prescribe particular notations to be used in the individual viewpoints. The viewpoint languages defined in the reference model are abstract languages in the sense that they define what concepts should be used, not how they should be represented. This lack of precise notations for expressing the different models involved in a multi-viewpoint specification of a system is a common feature for most enterprise architectural approaches, including the Zachman Framework, the "4+1" model, or the RM-ODP. These approaches were consciously defined in a notation- and representation-neutral manner to increase their use and flexibility. However, this makes more difficult, among other things, the development of industrial tools for modeling the viewpoint specifications, the formal analysis of the specifications produced, and the possible derivation of implementations from the system specifications.

In order to address these issues, ISO/IEC and the ITU-T started a joint project in 2004: "ITU-T Rec. X.906|ISO/IEC 19793: Information technology - Open distributed processing - Use of UML for ODP system specifications". This document (usually referred to as UML4ODP [10]) defines use of the Unified Modeling Language 2 (UML 2; ISO/IEC 19505), for expressing the specifications of open distributed systems in terms of the viewpoint specifications defined by the RM-ODP.

It defines a set of UML Profiles, one for each viewpoint language and one to express the correspondences between viewpoints, and an approach for structuring them according to the RM-ODP principles. The purpose of "UML4ODP" to allow ODP modelers to use the UML notation for expressing their ODP specifications in a standard graphical way; to allow UML modelers to use the RM-ODP concepts and mechanisms to structure their large UML system specifications according to a mature and standard proposal; and to allow UML tools to be used to process viewpoint specifications, thus facilitating the software design process and the enterprise architecture specification of large software systems.

In addition, ITU-T Rec. X.906 | ISO/IEC 19793 enables the seamless integration of the RM-ODP enterprise architecture framework with the Model-Driven Architecture (MDA) initiative from the OMG, and with the service-oriented architecture (SOA).

Applications

In addition, there are several projects that have used or currently use RM-ODP for effectively structuring their systems specifications:

- The COMBINE project[11]
- The Reference Architecture for Space Data Systems (RASDS)[12] From the Consultative Committee for Space Data Systems.
- Interoperability Technology Association for Information Processing (INTAP), Japan.[13]
- The Synapses European project.[14]

Notes and references

[1] A complete and updated list of references to publications related to RM-ODP (books, journal articles, conference papers, etc.) is available at the RM-ODP resource site (http://www.rm-odp.net/publications.html).

[2] In the same series as the RM-ODP are a number of other standards and recommendations for the specification and development of open and distributed system, for which RM-ODP provides a standardization framework:

- ITU-T Rec. X.950 | ISO/IEC 13235-1:1998, Trading function: Specification.
- ITU-T Rec. X.952 | ISO/IEC 13235-3:1998, Provision of Trading Function using OSI directory service.
- ITU-T Rec. X.920 | ISO/IEC 14750:1999, Interface Definition Language.
- ITU-T Rec. X.931 | ISO/IEC 14752:2000, Protocol support for computational interactions.
- ITU-T Rec. X.930 | ISO/IEC 14753:1999, Interface references and binding.
- ITU-T Rec. X.960 | ISO/IEC 14769:2001, Type repository function.
- ITU-T Rec. X.910 | ISO/IEC 14771:1999, Naming framework.
- ITU-T Rec. X.911 | ISO/IEC 15414:2002, Reference model - Enterprise language (http://www.joaquin.net/ODP/DIS_15414_X.911.pdf).
- ISO/IEC 19500-2:2003, General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).

[3] Copies of the RM-ODP family of standards can be obtained either from ISO (http://www.iso.ch) or from ITU-T (http://www.itu.int/). Parts 1 to 4 of the RM-ODP are available for from free download from ISO (http://isotc.iso.ch/livelink/livelink/fetch/2000/2489/Ittf_Home/PubliclyAvailableStandards.htm). All ODP-related ITU-T Recommendations, including X.9xx series, are freely available from the ITU-T (http://www.itu.int/rec/T-REC-X/en).

[4] There is also a very useful hyperlinked version (http://www.joaquin.net/ODP) of Parts 2 and 3 of the RM-ODP, together with an index to the Reference Model, made available in keeping with a resolution of the ISO council. The Table of Contents and Index were prepared by Lovelace Computing and are being made available by Lovelace Computing as a service to the standards community.

[5] Some resources related to the current version of | ITU-T X.906 | ISO/IEC 19793 "Use of UML for ODP systems specifications" (http://www.rm-odp.net/files/resources/LON-040_UML4ODP_IS/LON-040_UML4ODP_IS.pdf) are also available from the RM-ODP resource site (http://www.rm-odp.net). They include the UML Profiles of the five ODP viewpoints, the viewpoint metamodels, the GIF files for the ODP-specific icons, etc.

[6] ISO/IEC 10746-1 | ITU-T Rec. X.901 (http://standards.iso.org/ittf/PubliclyAvailableStandards/c020696_ISO_IEC_10746-1_1998(E).zip)

[7] ISO/IEC 10746-2 | ITU-T Rec. X.902 (http://standards.iso.org/ittf/PubliclyAvailableStandards/s018836_ISO_IEC_10746-2_1996(E).zip)

[8] ISO/IEC 10746-3 | ITU-T Rec. X.903 (http://standards.iso.org/ittf/PubliclyAvailableStandards/s020697_ISO_IEC_10746-3_1996(E).zip)

[9] ISO/IEC 10746-4 | ITU-T Rec. X.904 (http://standards.iso.org/ittf/PubliclyAvailableStandards/c020698_ISO_IEC_10746-4_1998(E).zip)

[10] http://www.lcc.uma.es/~av/download/UML4ODP_IS_V2.pdf

[11] COMBINE (http://www.opengroup.org/combine/overview.htm)

[12] Reference Architecture for Space Data Systems (RASDS) (http://public.ccsds.org/review/default.aspx)

[13] Interoperability Technology Association for Information Processing (INTAP) (http://www.net.intap.or.jp/e)

[14] The Synapses Project: a three-year project funded under the EU 4th Framework Health Telematics Programme (http://www.cs.tcd.ie/synapses/public/)

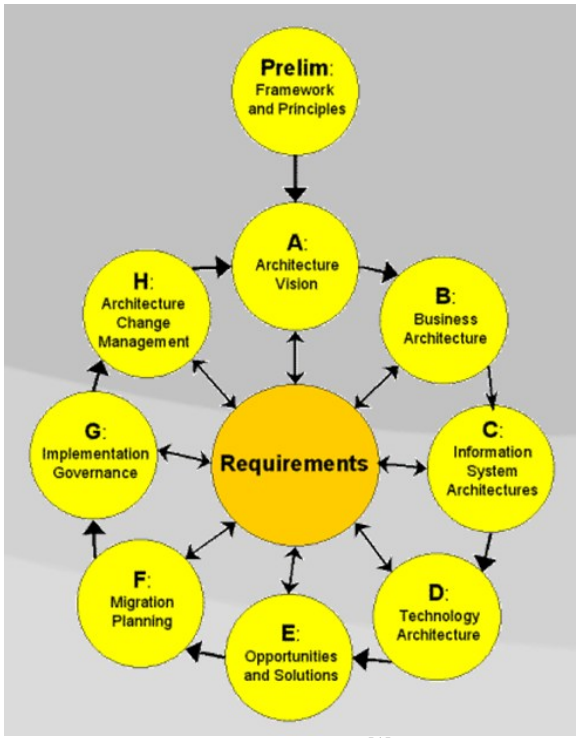
External links

- RM-ODP Resource site (http://www.rm-odp.net/)
- Open Distributed Processing - Reference Model (http://www.joaquin.net/ODP/)
- RM-ODP information at LAMS (http://lams.epfl.ch/reference/rm-odp), Swiss Federal Institute of Technology, Lausanne (EPFL), Switzerland.
- Official Record of the ANSA project (http://www.ansa.co.uk/)
- Computing Laboratory (http://www.cs.ukc.ac.uk/), University of Kent, Canterbury UK.
- FORMOSA (http://www.cs.stir.ac.uk/~kjt/research/formosa.html) (Formalisation of ODP Systems Architecture), University of Stirling, UK.
- Distributed and Cooperative Systems (http://www.lip6.fr/recherche/team.php?id=8&LANG=en), UMPC, Paris, France.
- ILR (http://www.infres.enst.fr/recherche/ILR/rapport.html), Networks and ComputerScience Department of ENST, Paris France.
- Distributed Systems Technology Center (http://archive.dstc.edu.au/AU/research_news/), Australia.

The Open Group Architecture Framework

The Open Group Architecture Framework (**TOGAF®**) is a framework for enterprise architecture which provides a comprehensive approach for designing, planning, implementing, and governing an enterprise information architecture. TOGAF is a registered trademark of The Open Group in the United States and other countries. [2]

TOGAF is a high level and holistic approach to design, which is typically modeled at four levels: Business, Application, Data, and Technology. It tries to give a well-tested overall starting model to information architects, which can then be built upon. It relies heavily on modularization, standardization and already existing, proven technologies and products.



TOGAF 8.1.1 ADM.[1]

Overview

An architecture framework is a set of tools which can be used for developing a broad range of different architectures.[3] It should:

- describe a method for defining an information system in terms of a set of building blocks
- show how the building blocks fit together
- contain a set of tools
- provide a common vocabulary
- include a list of recommended standards
- include a list of compliant products that can be used to implement the building blocks

TOGAF is such an architecture framework.

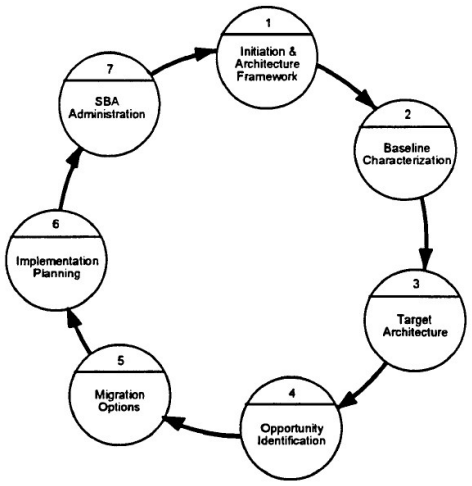
The ANSI/IEEE Standard 1471-2000 specification of architecture (of software-intensive systems) may be stated as: "the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution."

However TOGAF has its own view, which may be specified as either a "formal description of a system, or a detailed plan of the system at component level to guide its implementation", or as "the structure of components, their interrelationships, and the principles and guidelines governing their design and evolution over time."

History

TOGAF was developed by the Architecture Forum of The Open Group and has been continuously evolving since the mid-1990s. In 1995, the first version of TOGAF Version was presented, which was:

> "...based on the Technical Architecture Framework for Information Management (TAFIM). The US Department of Defense gave The Open Group explicit permission and encouragement to create TOGAF by building on the TAFIM, which itself was the result of many years of development effort and many millions of dollars of US Government investment."[5]



DoD Standards-Based Architecture Planning Process in TAFIM.[4]

TOGAF 7 ("Technical Edition") was published in December 2001. TOGAF 8 ("Enterprise Edition") was first published in December 2002 and republished in updated form as TOGAF 8.1 in December 2003, which was updated in November 2006 as TOGAF 8.1.1. According to The Open Group, as of February 2011, over 15,000 individuals are TOGAF Certified [6].[7]

The latest version is TOGAF 9, launched on 2 February 2009. An evolutionary development from TOGAF 8, TOGAF 9 [8][9] includes many new features including:

- Increased rigor, including a formal Content Metamodel that links the artifacts of TOGAF together
- Elimination of unnecessary differences
- Many more examples and templates

Additional guidelines and techniques include:

- A formal business-driven approach to architecture
- Business capability-based planning
- Guidance on how to use TOGAF to develop Security Architectures and SOAs

The Open Group provides TOGAF free of charge to organizations for their own internal noncommercial purposes.[10]

TOGAF topics

Enterprise architecture domains

TOGAF is based on four interrelated domains

Alternative enterprise architecture frameworks

- AGATE French Délégation Générale pour l'Armement Atelier de Gestion de l'ArchiTEcture des systèmes d'information et de communication.
- ArchiMate an open and independent modelling language for enterprise architecture
- ARCON - A Reference Architecture for Collaborative Networks - not focused on a single enterprise but rather on networks of enterprises [11][12]
- DoDAF United States Department of Defense Architectural Framework.
- CSC Catalyst [13] CSC Catalyst
- DYA framework Sogeti Framework.
- EIF European Interoperability Framework - Enterprise architecture at the level of EU Member States
- IDABC Interoperable Delivery (of European egovernment services to public) Administrations, Business and Citizens
- Integrated Architecture Framework (IAF) created by Capgemini.
- FEA United States Office of Management and Budget Federal Enterprise Architecture.
- MIKE2.0 (Method for an Integrated Knowledge Environment) which includes an enterprise architecture framework called SAFE (Strategic Architecture for the Federated Enterprise)
- MODAF United Kingdom Ministry of Defence Architectural Framework.
- Model-driven architecture (MDA) Object Management Group's Model Driven Architecture.
- OBASHI (The OBASHI Business & IT methodology and framework.
- The Operations Systems Computing Architecture (OSCAR), initially developed in 1986 by Bell Communications Research (the forerunner of the current Telcordia Technologies) to guide development of enterprise systems for the Regional Bell Operating Companies (RBOCs).
- PROMIS Framework [14] The PROMIS Enterprise Architecture Framework integrated into the EA tool EVA Netmodeler
- SABSA a comprehensive framework for Enterprise Security Architecture and Service Management.
- SAP Enterprise Architecture Framework is extension of TOGAF to better support Commercial off-the-shelf and Service-Oriented Architecture
- Zachman Framework IBM Framework from the 1980s.

References

[1] Stephen Marley (2003). Architectural Framework (http://aiwg.gsfc.nasa.gov/esappdocs/RPC/RPC_Workshop_Architecture_Framework.ppt). NASA /SCI. Retrieved 10 Dec 2008.

[2] TOGAF Trademark (http://blog.opengroup.org/2011/02/08/togafÂ®-trademark-success/)

[3] TOGAF Introduction (http://www.opengroup.org/architecture/togaf8-doc/arch/) The Open Group Architecture Framework. Accessed 22 Jan 2009.

[4] Department of Defense (1996). *Technical Architecture Framework for Information Management. Vol. 4*. April 1996

[5] The Open Group (2009). Welcome to TOGAF Version 9 -- The Open Group Architecture Framework. Retrieved on 2009-02-03 from http://www.opengroup.org/architecture/togaf9-doc/arch/.

[6] http://www.opengroup.org/togaf9/cert/cert_archlist-short.tpl

[7] 15,000 certifications (http://blog.opengroup.org/2011/02/08/togafÂ®-trademark-success/)

[8] http://www.opengroup.org/togaf/

[9] TOGAF 9.1 White Paper An Introduction to TOGAF Version 9.1 http://www.opengroup.org/togaf/

[10] The Open Group (2011). TOGAF® Version 9 - Download. Architecture Forum. Retrieved on 2011-11-17 from http://www.opengroup.org/architecture/togaf9/downloads.htm.

[11] L.M. Camarinha-Matos, H. Afsarmanesh, Collaborative Networks: Reference Modeling, Springer, 2008.
[12] L.M. Camarinha-Matos, H. Afsarmanesh, On reference models for collaborative networked organizations, International Journal Production Research, Vol 46, Nº 9, May 2008, pp 2453–2469.
[13] http://www.csc.com/delivery_excellence/ds/11388-csc_catalyst
[14] http://pro-mis.com/framework.html

External links

- Official website (http://www.togaf.info/)
- TOGAF 9 Online (http://www.opengroup.org/architecture/togaf9-doc/arch/)
- TOGAF 8.1.1 Online (http://www.opengroup.org/architecture/togaf8-doc/arch/)
- IBM developerWorks: Understand The Open Group Architecture Framework (TOGAF) and IT architecture in today's world (http://www-128.ibm.com/developerworks/ibm/library/ar-togaf1/) (February 2006)
- Developer.com: TOGAF: Establishing Itself As the Definitive Method for Building Enterprise Architectures in the Commercial World (http://www.developer.com/design/article.php/3374171) (June 2004)
- TOGAF or not TOGAF: Extending Enterprise Architecture beyond RUP (http://www-128.ibm.com/developerworks/rational/library/jan07/temnenco/index.html) (January 2007)
- Practical advice: How to bring TOGAF to life (http://togaforblunder.blogspot.com/) (October 2007)
- Togaf Modeling using UML and BPMN (http://www.togaf-modeling.org/) (May 2010)

Commercial frameworks

Integrated Architecture Framework

The **Integrated Architecture Framework** (IAF) is an enterprise architecture framework that covers business, information, information system and technology infrastructure.[1][2]

This framework has been developed by Capgemini since the 1990s, from the experience of practicing architects on projects for clients across the group. The first version was released in 1996 and was based on the Zachman Framework and Spewaks ideas about Enterprise Architecture Planning.[3]

The Integrated Architecture Framework is:

- A comprehensive framework to deliver market-leading solutions
- Adaptable to the specific needs of an organization
- Scalable from individual projects to enterprise-wide transformation
- A recognized architecture method in The Open Group's IT Architecture Certification program (ITAC).[4]

The Integrated Architecture Framework has evolved based on real-world experience, and continues to provide strong focus on the need to understand business requirements and drivers, and the need for all aspects of the architecture and all architectural decisions to be traceable back to these business priorities.

References

[1] http://www.capgemini.com/insights-and-resources/by-publication/enterprise_business_and_it_architecture_and_the_integrated_architecture_framework/

[2] van't Wout, J., Waage, M., Hartman, H., Stahlecker, M., Hofman, A. (2010). "The Integrated Architecture Framework Explained" - http://www.springer.com/business+&+management/business+information+systems/book/978-3-642-11517-2

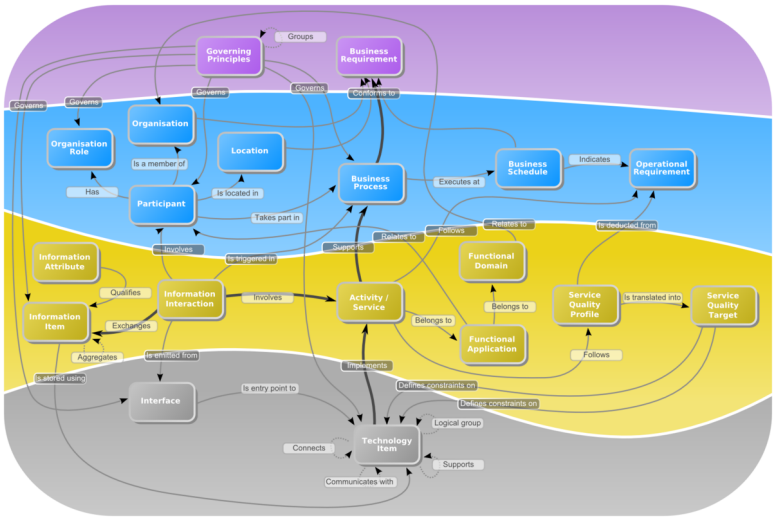
[3] Jaap Schekkerman (2003). *How to Survive in the Jungle of Enterprise Architecture Frameworks*. page 139-144.

[4] ITAC - http://www.opengroup.org/itac/, http://www.opengroup.org/itac/cert/methods.tpl

CLEAR Framework for Enterprise Architecture

CLEAR is an Enterprise Architecture meta-model, framework and method. CLEAR stands for *Comprehensive, Landscaped, Enterprise Architecture Representation*.

CLEAR was developed by Atos Origin as a Enterprise Architecture framework, meta-model and method aiming at facilitating alignment between an Enterprise's business objectives and the technologies it deploys to realise them.



Landscapes, Artefacts and Relationships in the modelling part of the CLEAR EA Framework

Overview

CLEAR borrows heavily from the Zachman Framework for Enterprise Architecture and the TOGAF framework specification. CLEAR is divided into four landscapes which contain artefacts. These landscapes are:

- The *Environmental* landscape which contains the Business Objectives or *Contextual* landscape
- The *Business* landscape which contains the *Conceptual* Business Architecture
- The *Alignment* Landscape which contains a functional architecture and services catalogue
- The *Technology* Landscape which contains all of the technical components necessary to deliver the business services (Software, Hardware and Infrastructure elements)

CLEAR's *Services Oriented* Alignment layer can be used to provide the specification for a Service-oriented architecture approach.

Meta-model

The CLEAR meta-model contains the above landscapes and artefacts together with a Taxonomy and Ontology to provide traceability and consistency across the architecture.

Framework

CLEAR contains an Information Technology Governance framework and processes for Enterprise Architecture. These are largely derived from the ITIL specification for service management.

Method

CLEAR contains a method for building the Enterprise Architecture itself. This is loosely based on the TOGAF ADM but is claimed as being more focused on the solution of business problem areas rather than on building the Architecture as an end in itself.

Alternative Enterprise Architecture Frameworks

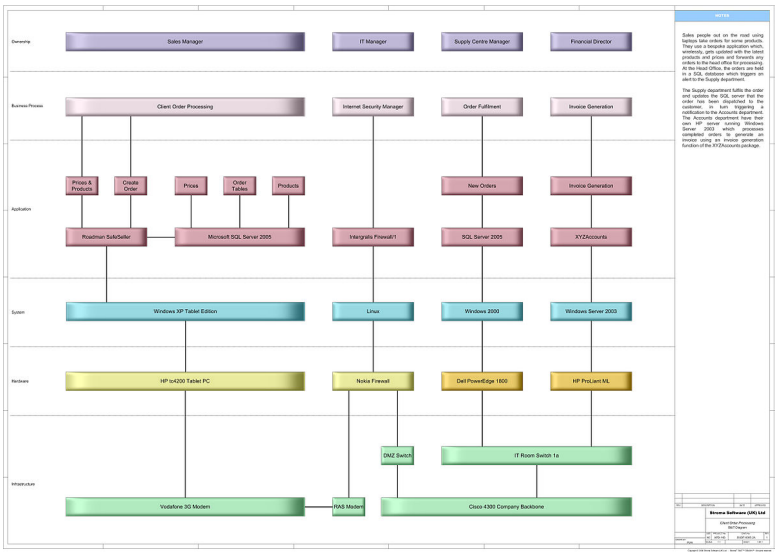
- Zachman Framework
- DoDAF
- FEAF
- MODAF
- AGATE
- TOGAF

OBASHI

The **OBASHI methodology** provides a framework and method for capturing, illustrating and modeling the relationships, dependencies and dataflows between business and Information technology (IT) assets and resources in a business context.

It is a formal and structured way of communicating the logical and physical relationships and dependencies between IT assets and resources (Ownership, Business Processes, Applications, Systems, Hardware, and Infrastructure) to define the business services of a modern enterprise.

The name *OBASHI* is a licensed trademark of OBASHI Ltd.



A Business and IT (B&IT) diagram built using the OBASHI Framework.

Core Principle

OBASHI is based around a core principle: that IT exists for one reason, namely, to manage the flow of data between business assets.

Business resources (which include people) and IT assets are either providers of data, consumers of data or provide the conduit through which the data can flow.

The role of IT is to support, process and optimise the flow of data to maximise business/organisational performance.

The Origins of OBASHI

The **OBASHI framework** was invented by Fergus Cloughley and Paul Wallis of Stroma Software Ltd. during late 2001, following their collaboration on a project to help plant managers visualise and understand how and why IT assets supported business services within British Petroleum, Grangemouth, Scotland.

The subsequent **OBASHI methodology** was born out of the need for business professionals to easily understand the dollar per second value of dataflow that supports their business services in a simple and meaningful way so accurate and better informed operational and strategic decisions could be made.

Cloughley and Wallis recognized that by developing a methodology around the OBASHI framework, the existing methods for costing and valuing the flow of data in the Oil & Gas / Process Control industry could be made universally appllicable to flows of data in all sectors.

The methodology entails capturing, documenting, modeling, analysing, simulating and optimizing the cost / value of the flow of data between assets and business resources.

DCS - CAD in Process control

The Oil & Gas, Chemical, Pharmaceutical, Power Generation, Water, Food and Beverage, Cement, Steelmaking and Paper industries have relied on Distributed Control Systems (DCS) to manage and control their Manufacturing and Process Control systems.

The flow of data for I/O control purposes is fundamental to the safe and efficient function of plant operations. A clear and concise understanding of the cause and effect of this data flow is a prerequisite to plant operations, business optimization and Health and Safety governance.

This clear and concise understanding is supported by a rigorous approach to documentation, namely: Computer Aided Design (CAD) diagrams & models; Piping and Instrumentation diagrams (P&IDs); Process Flow Diagrams (PFDs); and Cause and Effect diagrams.

Many other add-on technologies can be linked to the flows of plant data to support and optimise the plant running conditions, for example: Chemical Process modeling, Computer simulation, Process optimization and plant maintenance management systems.

The **OBASHI Framework** and **methodology** were developed to mimic this rigorous approach and provide contextual documentation to support safe and efficient IT & Business operational practices.

Data flows in the Oil and Gas Industries

Computer models are used within manufacturing and process industries to control and simulate the operation of the plant. These models derive real-time input from digital sensors attached to physical equipment throughout the manufacturing process and are built to understand the contextual relationship between the assets and process flow.

By linking the model to the commodities markets the real costs/values of flow can be displayed, monitored and trended as dollars per second, enabling the transposition from tonnes per hour. In turn, this enables business processes to be optimised around value, and each asset's contribution to the cost/value of the flow be evaluated in financial terms.

Data flows in the modern business

The contextual understanding of the relationships between physical assets and flows is used within OBASHI to model the flow of data between business services and IT equipment.

Within the OBASHI methodology, business resources and IT assets are regarded as either providers of data, consumers of data, or provide the conduit through which the data can flow.

People provide and consume data daily, as do applications and systems. Hardware and cables act as conduits through which data flows: between desks, through office and corporate networks, across the internet, through deep sea cables and via satellites.

Across all businesses, the equivalents of the pipes, valves, pumps, meters and sensors of the oil and gas industry are the people, hubs, cables, routers, servers, and desktops through which data flows.

By utilising comparable contextual relationships the OBASHI methodology enables dataflow to be analysed in a similar manner to that of process flow within the manufacturing industry, which is central to business performance optimisation.

Characteristics

The **OBASHI Methodology** models the enterprise/an organisation in six horizontal layers. The layers provide a framework (**the OBASHI framework**) for organising individual elements that represent individual Business or IT assets and resources. The layers are:

- Ownership
- Business Process
- Application
- System
- Hardware
- Infrastructure

Placing the elements above or below each other within the framework signifies a relationship between the elements. For example, placing an Owner element above a Business Process element signifies that the business processes belongs to that owner. Placing a business function above an application signifies that the process uses that application etc...

Elements can be connected on the diagram to denote a physical relationship, such as the connection between a hardware element and an infrastructure element. Dependencies can also be documented on the diagram to show explicit non-obvious relationships between elements, such as the reliance of a business process on a third party resource.

Each element can be referenced to supporting documentation to provide a supporting context for that element.

The flow of data (dataflows) can be superimposed on the diagrams to depict a sequence of elements required to support a business service.

The combination of one or more OBASHI diagrams form a contextual model for analysis.

The OBASHI Framework

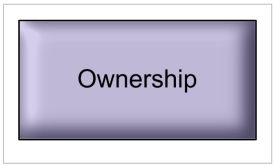
These Layers provide the framework for organising the elements that represent individual Business or IT assets / resources.

The six layers are:

- **O**wnership
- **B**usiness Process
- **A**pplication
- **S**ystem
- **H**ardware
- **I**nfrastructure

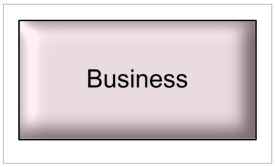
They are collectively known as OBASHI

Ownership Layer



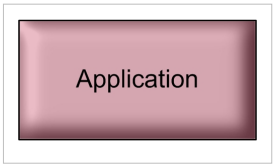
The Ownership Layer contains elements representing the person(s) or group(s) that 'owns', or is responsible for, business processes portrayed in the Business Layer. Ownership elements can be positioned beneath other ownership elements to create a hierarchy of owners. Example owners could be: *Accountancy*, *Planning Manager*, *Logistics*, *New York*, *Purchasing Officer* and *Environmental Health*.

Business Layer



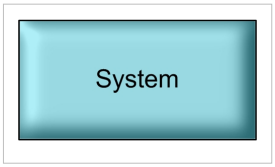
The Business Layer contains elements representing the business processes or functions that are being used by the 'Owner(s)'. These elements are positioned under their appropriate 'Owner'. Examples could be: *Monthly Balance*, *Sales Transactions*, *Tank Stock Management*, *Production Data* and *Capture Budgeting*.

Application Layer



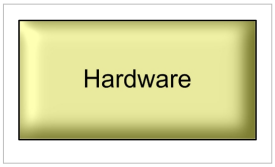
The Application Layer contains elements representing software applications. These are positioned beneath the business processes that utilise them. Examples could include: *Excel*, *Oracle*, *Sage*, *SAP* and *PeopleSoft*.

System Layer



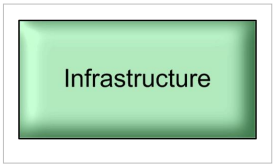
The System Layer contains elements representing the operating systems on which the applications run. These elements are positioned beneath the appropriate applications. Examples could be: *Windows XP*, *Unix*, *Solaris*, *Linux* and *Vista*.

Hardware Layer



The Hardware Layer contains elements representing the computer hardware on which the operating systems run. These elements are positioned beneath the appropriate operating systems. Examples could be: *Workstations*, *Servers*, *Laptops*, *Tablet PCs*, and *Mainframes*.

Infrastructure Layer



The Infrastructure Layer contains elements representing the network infrastructure into which the hardware is connected. Infrastructure elements can be positioned beneath other infrastructure elements to create a hierarchy that supports the business. Examples could be: *Switches*, *Routers*, *Multiplexers*, *Bridges* and *Hubs*.

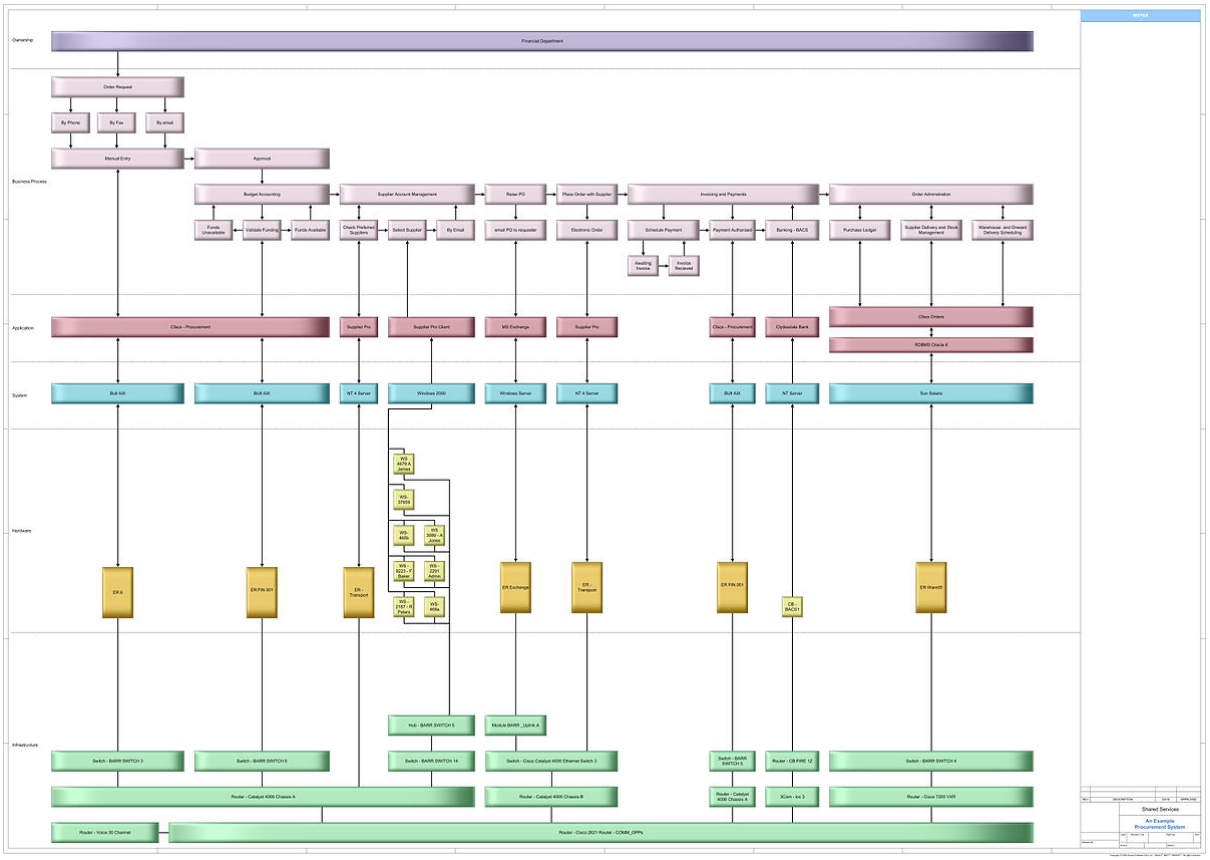
B&IT Diagrams

The **OBASHI framework** is used to create Business and IT *(B&IT)* diagrams. A B&IT diagram is a diagrammatic representation of the logical and physical relationships (connectivity) between an organisation's IT assets and resources and the business operations which they support.

A B&IT diagram is made up of elements. Individual elements represent individual business and IT assets and resources.

By employing the OBASHI framework, B&IT diagrams are able to accurately depict the complex inter-relationships and dependencies of business processes, IT resources and dataflows in an easy-to-understand visual format.

Through the use of a tool which supports the OBASHI Framework a repository of elements and relationships can be established. By using a graphical interface to create the B&IT diagrams the tool can build an interactive model of these relationships, with the B&IT diagram acting as a dynamic interface.



Behind each element information can be stored within the repository: business, financial and/or technical. This data may be captured manually, or automatically from data held within existing systems. This information can then be viewed, manipulated and analysed within its business context.

Rules

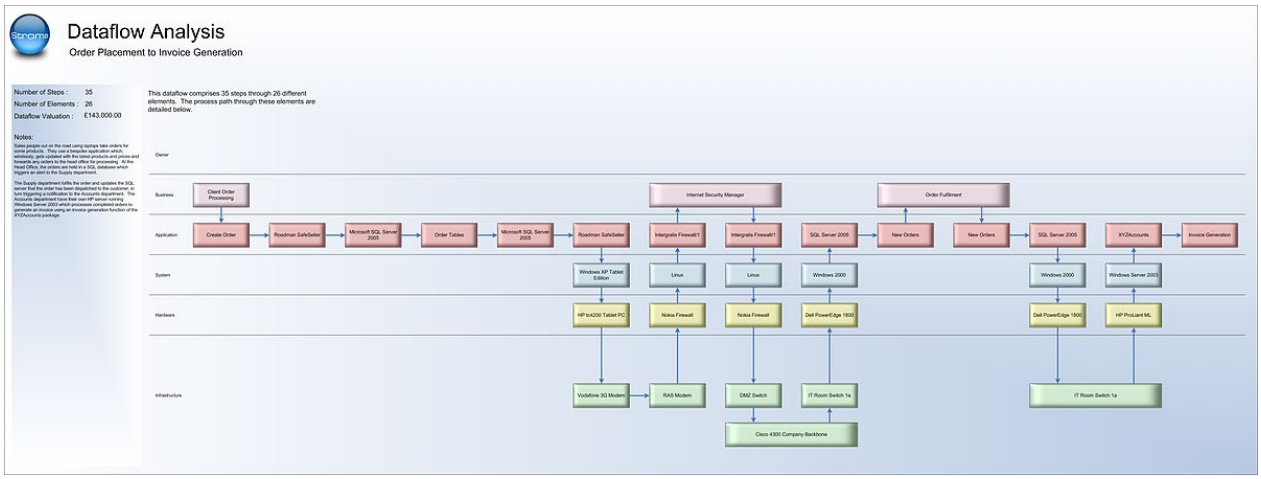
The **OBASHI Framework** comprises the following rules which govern the implicit and explicit relationships between elements.

1. An element placed beneath or above another element has an implicit relationship with that element.
2. All elements within the same layer have an implicit relationship to each other.
3. Connected elements have an explicit relationship to each other, with the following rules governing connectivity:
    1. A connection is a bi-directional relationship
    2. An Infrastructure element may be connected to one or more Infrastructure or Hardware elements.
    3. A Hardware element may be connected to one or more Infrastructure or System elements.
    4. A System element may be connected to one or more Hardware or Application elements.
    5. An Application element may be connected to one or more System, Application or Business elements.
    6. A Business element may be connected to one or more Application, Business or Owner elements.
    7. An Owner element may be connected to one or more Business or Owner elements.
4. A dependency is a uni-directional relationship i.e. element X may be dependent on element Y, but element Y might not be dependent on element X.
5. An element may have one or more instances within a layer.
6. An element can exist on more than one OBASHI diagram.
7. A dataflow comprises two or more connected elements.
8. A dataflow can contain one or more dataflows, enabling a hierarchy of dataflows.
9. A dataflow may span multiple Business and IT diagrams.

The Dataflow Analysis View (DAV)

The **OBASHI Dataflow Analysis View** (*DAV*) is a graphical and statistical representation of all the business and IT resources, and attributed financial values, that support an individual data flow. A **DAV** illustrates to business professionals how and why IT systems interact with day to day business processes.

All of the physical assets required for the logical data flow to exist are documented as a sequence. A sequence shows all of the IT and Business assets and resources involved in the data flow.



The **DAV** enables cost/value statistics to be generated to understand the contribution IT assets make to the business. Analysis of the **DAV** can highlight vulnerabilities, mis-alignment and areas for consolidation.

Fields of use

The OBASHI Methodology can be used in the following fields:

- Availability
- Business Architecture
- Business Continuity
- Business Process Management (BPM)
- Business Service Management
- Business Technology Optimisation (BTO)
- Change Management
- CHAZOP (Computer Hazardous Operation Analysis)
- COBIT
- Data Center Management
- Enterprise Architecture (EA)
- Governance & Auditing
- Grid Computing
- Information Architecture (IA)
- Enterprise Information Security Architecture (EISA)
- Infrastructure Management
- IT Service Management
- IT Security
- ITIL
- On Demand/Utility Computing
- Performability
- Risk Management
- Service Oriented Architecture (SOA)

- SOX
- Technical Architecture

References

- APMG-International (2010), The OBASHI Methodology. The Stationery Office ISBN 978-0-11-706857-5.
- Neil McNaughton, Editor, Oil Information Technology Journal [1], Oilit.com. *Business and IT Mapping, Petroleum Exploration Society Great Britain Data Management Conference, Technology Watch Report*, London (December 11, 2007)
- Karl Jeffery, Editor, *Seeing IT in a Business Context* Digital Energy Journal, Feature Article, (August 7, 2007)

External links

- The official OBASHI Website [2]
- The APMG-International OBASHI Webpage [3]

References

[1] http://www.oilit.com
[2] http://www.obashi.co.uk/
[3] http://www.apmg-international.com/APMG-UK/OBASHI/OBASHIHome.asp

Information Framework

This article is about one specific framework proposed in the 1990s

The **Information FrameWork (IFW)** is a Enterprise Architecture framework, originally conceived by Roger Evernden[1][2] in 1996.[3] The Information FrameWork is a family of data, process and object models to help financial institutions transform cross-enterprise architectures.[4] It used to classify enterprise architecture deliverables in much the same way as the Zachman Framework.

In 1987 had John Zachman proposed the Zachman Framework to describe Information Architecture with the six concepts: The *what* related to data, *how* related to process, *where* related to network and location, *who* related to actors and people, *when* related to time, and at last *why* related to motivation. According to Everden (1996) "research in the 1990s had indicated a tremendous need for integration of these descriptions into a unified paradigm and for the creation of Computer-aided software engineering (CASE) tools to support Information Architecture modelling".[3][5]

The Information Framework (IFW) is based around a number of architecture domain views, with three views, in ten columns:

- Organisation Architecture View (Structure, Strategy, Skills),
- Business Architecture View (Data, Business Functions, Workflow, Solutions),
- Technology Architecture View (Applications, Networks, Systems).

Each Architecture Domain is further viewed in terms of the levels of detail (A, B, C, C', D).

The Information FrameWork according to Everden (2003), a "brain dump his thoughts on information architecture, based on using the Information Framework, he developed at IBM.[6] The objectives and scope of IFW are broader, than that of the original Zachman Framework, based on experiences from within the financial services industry.

References

[1] Roger Evernden - a brief autobiography (http://www.evernden.net/content/rb.htm) Accessed 17 Jan 2009.
[2] Roger Evernden (http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/e/Evernden:Roger.html) List of publications from the DBLP Bibliography Server. Accessed 17 Jan 2009.]
[3] Roger Evernden (1996). "The Information FrameWork". In: *IBM Systems Journal* 35(1): 37-68 (1996)
[4] Information FrameWork (http://www-03.ibm.com/industries/financialservices/us/detail/component/I803938H46550Z52.html) at IBM.com. Accessed 17 Jan 2009.
[5] Shouhong Wang (2001). *Analyzing Business Information Systems* p.150.
[6] Roger Evernden (2003). *Information First*. p 201.

Further reading

- Danny Greefhorst et al. (2006). " The many faces of architectural descriptions (http://www.cs.vu.nl/~hans/publications/y2006/facesISF.pdf)" In: *Inf Syst Front* (2006) 8:103–113
- Richard Martin (2006). " Formalization of Multi-level Zachman Frameworks (http://www.cs.indiana.edu/pub/techreports/TR522.pdf)" TECHNICAL REPORT NO. 522 COMPUTER SCIENCE DEPARTMENT, INDIANA UNIVERSITY.

External links

- Information FrameWork (http://www-03.ibm.com/industries/financialservices/us/detail/component/I803938H46550Z52.html) at IBM.com

Zachman Framework

The **Zachman Framework** is an Enterprise Architecture framework for enterprise architecture, which provides a formal and highly structured way of viewing and defining an enterprise. It consists of a two dimensional classification matrix based on the intersection of six communication questions (What, Where, When, Why, Who and How) with six rows according to reification transformations.[1]

	Why	How	What	Who	Where	When
Contextual	Goal List	Process List	Material List	Organisational Unit & Role List	Geographical Locations List	Event List
Conceptual	Goal Relationship	Process Model	Entity Relationship Model	Organisational Unit & Role Relationship Model	Locations Model	Event Model
Logical	Rules Diagram	Process Diagram	Data Model Diagram	Role Relationship Diagram	Locations Diagram	Event Diagram
Physical	Rules Specification	Process Function Specification	Data Entity Specification	Role Specification	Location Specification	Event Specification
Detailed	Rules Details	Process Details	Data Details	Role Details	Location Details	Event Details

The Zachman Framework of Enterprise Architecture

The Zachman Framework is not a methodology in that it does not imply any specific method or process for collecting, managing, or using the information that it describes.[2] The Framework is named after its creator John Zachman, who first developed the concept in the 1980s at IBM. It has been updated several times since.[3]

The Zachman "Framework" is a schema for organizing architectural artifacts (in other words, design documents, specifications, and models) that takes into account both whom the artifact targets (for example, business owner and builder) and what particular issue (for example, data and functionality) is being addressed.[4]

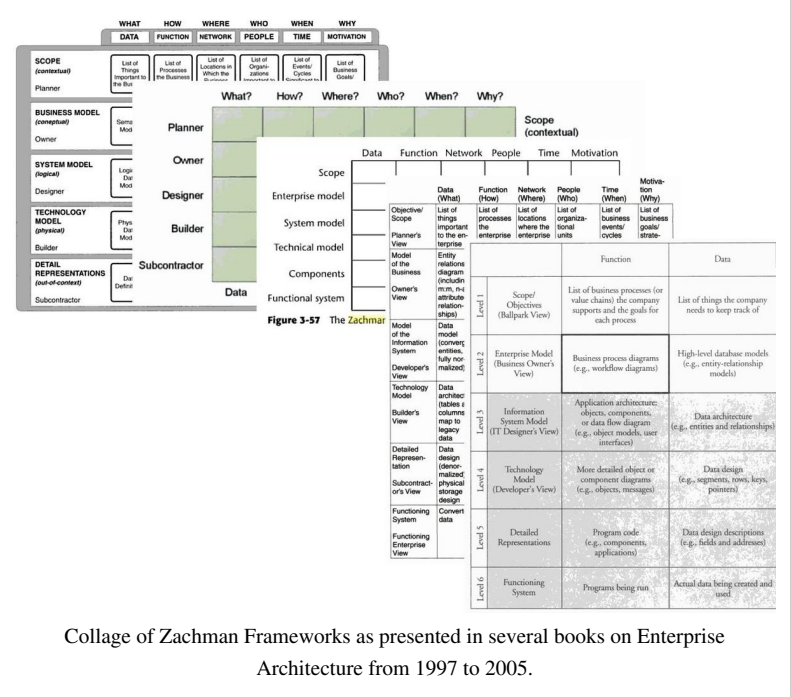
Overview

The term "Zachman Framework" has multiple meanings. It can refer to any of the frameworks proposed by John Zachman:

- The initial framework, named *A Framework for Information Systems Architecture*, by John Zachman published in an 1987 article in the IBM Systems journal.[5]
- The *Zachman Framework for Enterprise Architecture*, an update of the 1987 original in the 1990s extended and renamed .[6]
- One of the later versions of the Zachman Framework, offered by Zachman International as industry standard.

In other sources the Zachman Framework is introduced as a framework, originated by and named after John Zachman, represented in numerous ways, see image. This framework is explained as, for example:

- a framework to organize and analyze data,[7]
- a framework for enterprise architecture.[8]
- a classification system, or classification scheme[9]
- a matrix, often in a 6x6 matrix format
- a two-dimensional model[10] or an analytic model.
- a two-dimensional schema, used to organize the detailed representations of the enterprise.[11]



Collage of Zachman Frameworks as presented in several books on Enterprise Architecture from 1997 to 2005.

Beside the frameworks developed by John Zachman numerous extensions and or applications have been developed, which are also sometimes called Zachman Frameworks.

The Zachman Framework summarizes a collection of perspectives involved in enterprise architecture. These perspectives are represented in a two-dimensional matrix that defines along the rows the type of stakeholders and with the columns the aspects of the architecture. The framework does not define a methodology for an architecture. Rather, the matrix is a template that must be filled in by the goals/rules, processes, material, roles, locations, and events specifically required by the organization. Further modeling by mapping between columns in the framework identifies gaps in the documented state of the organization.[12]

The framework is a simple and logical structure for classifying and organizing the descriptive representations of an enterprise. It is significant to both the management of the enterprise, and the actors involved in the development of enterprise systems.[13] While there is no order of priority for the columns of the Framework, the top-down order of the rows is significant to the alignment of business concepts and the actual physical enterprise. The level of detail in the Framework is a function of each cell (and not the rows). When done by IT the lower level of focus is on information technology, however it can apply equally to physical material (ball valves, piping, transformers, fuse boxes for example) and the associated physical processes, roles, locations etc. related to those items.

History

In the 1980s John Zachman had been involved at IBM in the development of Business System Planning (BSP), a method for analyzing, defining and designing an information architecture of organizations. In 1982 Zachman[14] had already concluded that these analyses could reach far beyond automating systems design and managing data into the realms of strategic business planning and management science in general. It may be employed in the (in that time considered more esoteric) areas of enterprise architecture, data-driven systems design, data classification criteria, and more.[14]

Information Systems Architecture Framework

In the 1987 article "A Framework for Information Systems Architecture"[15] Zachman noted that the term "architecture" was used loosely by information systems professionals, and meant different things to planners, designers, programmers, communication specialists, and others.[16] In searching for an objective, independent basis upon which to develop a framework for information systems architecture, Zachman looked at the field of classical architecture, and a variety of complex engineering projects in industry. He saw a similar approach and concluded that architectures exist on many levels and involves at least three perspectives: raw material or data, function of processes, and location or networks.[16]

	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*
Objective/Scope (contextual) *Role: Planner*	List of things important in the business	List of Business Processes	List of Business Locations	List of important Organizations	List of Events	List of Business Goal & Strategies
Enterprise Model (conceptual) *Role: Owner*	Conceptual Data/ Object Model	Business Process Model	Business Logistics System	Work Flow Model	Master Schedule	Business Plan
System Model (logical) *Role:Designer*	Logical Data Model	System Architecture Model	Distributed Systems Architecture	Human Interface Architecture	Processing Structure	Business Rule Model
Technology Model (physical) *Role:Builder*	Physical Data/Class Model	Technology Design Model	Technology Architecture	Presentation Architecture	Control Structure	Rule Design
Detailed Reprentation (out of context) *Role: Programmer*	Data Definition	Program	Network Architecture	Security Architecture	Timing Definition	Rule Speculation
Functioning Enterprise *Role: User*	Usable Data	Working Function	Usable Network	Functioning Organization	Implemented Schedule	Working Strategy

Simple example of the 1992 Framework.

The Information Systems Architecture is designed to be a classification schema for organizing architecture models. It provides a synoptic view of the models needed for enterprise architecture. Information Systems Architecture does not define in detail what the models should contain, it does not enforce the modeling language used for each model, and it does not propose a method for creating these models.[17]

Extension and formalization

In the 1992 article "Extending and Formalizing the Framework for Information Systems Architecture" John F. Sowa and John Zachman present the framework and its recent extensions and show how it can be formalized in the notation of conceptual graphs.[18] Also in 1992:

> John Zachman's co-author John Sowa proposed the additions of the Scope perspective of the 'planner' (bounding lists common to the enterprise and its environment) and the Detailed Representation perspective of the 'sub-contractor' (being the out of context vendor solution components). The Who, When and Why columns were brought into public view, the notion of the four levels of metaframeworks and a depiction of integration associations across the perspectives were all outlined in the paper. Keri Anderson Healey assisted by creating a model of the models (the framework metamodel) which was also included in the article.
>
> —Stan Locke, *Enterprise Convergence in Our Lifetime, from THE ENTERPRISE NEWSLETTER*[19]

Later during the 1990s[19]

- Methodologists like Clive Finkelstein refocused on the top two framework rows which he labeled Enterprise Engineering and has one of the most successful methods for converging the business needs with information

engineering implementation, and determining a logical build sequence of the pieces.

Framework for enterprise architecture

In the 1997 paper "Concepts of the Framework for Enterprise Architecture" Zachman said that the framework should be referred to as a "Framework for Enterprise Architecture", and should have from the beginning. In the early 1980s however, according to Zachman, there was "little interest in the idea of Enterprise Reengineering or Enterprise Modeling and the use of formalisms and models was generally limited to some aspects of application development within the Information Systems community".[20]

In 2008 Zachman Enterprise introduced the Zachman Framework: The Official Concise Definition as a new Zachman Framework standard.

Extended and modified frameworks

Since the 1990s several extended frameworks have been proposed, such as:

- Matthew & McGee (1990)[21] extended the three initial perspectives "what", "how" and "where", to event (the "when"), reason (the "why") and organization (the "who").[16]
- Evernden (1996) presented an alternative Information FrameWork.
- The Integrated Architecture Framework developed by Capgemini since 1996.[22]
- Vladan Jovanovic et all (2006) presents a Zachman Cube, an extended of the Zachman Framework into a multidimensional Zachman's Cube.[23]

Zachman Framework topics

Concept

The basic idea behind the Zachman Framework is that the same complex thing or item can be described for different purposes in different ways using different types of descriptions (e.g., textual, graphical). The Zachman Framework provides the thirty-six necessary categories for completely describing anything; especially complex things like manufactured goods (e.g., appliances), constructed structures (e.g., buildings), and enterprises (e.g., the organization and all of its goals, people, and technologies). The framework provides six different transformations of an abstract idea (not increasing in detail, but transforming) from six different perspectives.[24]

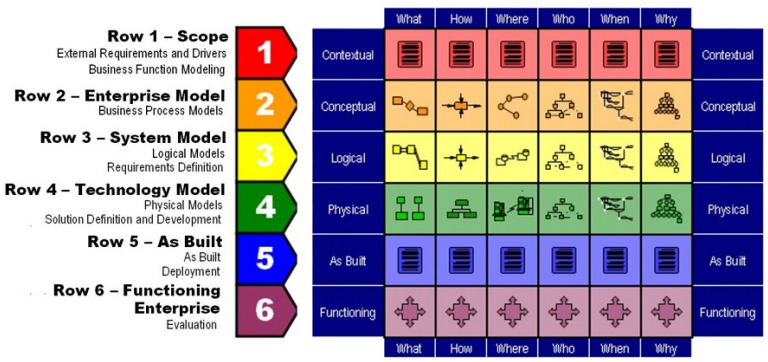
It allows different people to look at the same thing from different perspectives. This creates a holistic view of the environment, an important capability illustrated in the figure.[25]

Views of Rows

Each row represents a total view of the solution from a particular perspective. An upper row or perspective does not necessarily have a more comprehensive understanding of the whole than a lower perspective. Each row represents a distinct, unique perspective; however, the deliverables from each perspective must provide sufficient detail to define the solution at the level of perspective and must translate to the next lower row explicitly.[26]

Each perspective must take into account the requirements of the other perspectives and the restraint those perspectives impose. The constraints of each perspective are additive. For example, the constraints of higher rows affect the rows below. The constraints of lower rows can, but do not necessarily affect the higher rows. Understanding the requirements and constraints necessitates communication of knowledge and understanding from perspective to perspective. The Framework points the vertical direction for that communication between perspectives.[26]

In the 1997 Zachman Framework the rows are described as follows:[26]



The Veterans Affairs Zachman Framework with an explanation of its rows.[27][28]

- *Planner's View* (Scope) - The first architectural sketch is a "bubble chart" or Venn diagram, which depicts in gross terms the size, shape, partial relationships, and basic purpose of the final structure. It corresponds to an executive summary for a planner or investor who wants an overview or estimate of the scope of the system, what it would cost, and how it would relate to the general environment in which it will operate.
- *Owner's View* (Enterprise or Business Model) - Next are the architect's drawings that depict the final building from the perspective of the owner, who will have to live with it in the daily routines of business. They correspond to the enterprise (business) models, which constitute the designs of the business and show the business entities and processes and how they relate.
- *Designer's View* (Information Systems Model) - The architect's plans are the translation of the drawings into detail requirements representations from the designer's perspective. They correspond to the system model designed by a systems analyst who must determine the data elements, logical process flows, and functions that represent business entities and processes.
- *Builder's View* (Technology Model) - The contractor must redraw the architect's plans to represent the builder's perspective, with sufficient detail to understand the constraints of tools, technology, and materials. The builder's plans correspond to the technology models, which must adapt the information systems model to the details of the programming languages, input/output (I/O) devices, or other required supporting technology.
- *Subcontractor View* (Detailed Specifications) - Subcontractors work from shop plans that specify the details of parts or subsections. These correspond to the detailed specifications that are given to programmers who code individual modules without being concerned with the overall context or structure of the system. Alternatively, they could represent the detailed requirements for various commercial-off-the-shelf (COTS), government off-the-shelf (GOTS), or components of modular systems software being procured and implemented rather than built.
- *Actual System View* or The Functioning Enterprise

Focus of Columns

In summary, each perspective focuses attention on the same fundamental questions, then answers those questions from that viewpoint, creating different descriptive representations (i.e., models), which translate from higher to lower perspectives. The basic model for the focus (or product abstraction) remains constant. The basic model of each column is uniquely defined, yet related across and down the matrix.[26] In addition, the six categories of enterprise architecture components, and the underlying interrogatives that they answer, form the columns of the Zachman Framework and these are:[24]

1. The data description — What
2. The function description — How
3. The Network description — Where
4. The people description — Who
5. The time description — When
6. The motivation description — Why

In Zachman's opinion, the single factor that makes his framework unique is that each element on either axis of the matrix is explicitly distinguishable from all the other elements on that axis. The representations in each cell of the matrix are not merely successive levels of increasing detail, but actually are different representations — different in context, meaning, motivation, and use. Because each of the elements on either axis is explicitly different from the others, it is possible to define precisely what belongs in each cell.[24]

Models of Cells

The kinds of models or architectural descriptive representations are made explicit at the intersections of the rows and columns. An intersection is referred to as a cell. Because a cell is created by the intersection of a perspective and a focus, each is distinctive and unique. Since each cell is distinctive and unique, the contents of the cell are normalized and explicit per the perspective's focus.[26]

The cell descriptions in the table itself uses general language for a specific set of targets. Below the focus of each cell in this particular Zachman Framework is explained:

	Why	How	What	Who	Where	When
Contextual	Goal List	Process List	Material List	Organizational Unit & Role List	Geographical Locations List	Event List
Conceptual	Goal Relationship	Process Model	Entity Relationship Model	Organizational Unit & Role Rel. Model	Locations Model	Event Model
Logical	Rules Diagram	Process Diagram	Data Model Diagram	Role Relationship Diagram	Locations Diagram	Event Diagram
Physical	Rules Specification	Process Function Specification	Data Entity Specification	Role Specification	Location Specification	Event Specification
Detailed	Rules Details	Process Details	Data Details	Role Details	Location details	Event Details

Current view of the Zachman Framework.

Contextual

1. (Why) Goal List – primary high level organization goals
2. (How) Process List – list of all known processes
3. (What) Material List – list of all known organizational entities
4. (Who) Organizational Unit & Role List – list of all organization units, sub-units, and identified roles
5. (Where) Geographical Locations List – locations important to organization; can be large and small
6. (When) Event List – list of triggers and cycles important to organization

Conceptual

1. (Why) Goal Relationship Model – identifies hierarchy of goals that support primary goals
2. (How) Process Model – provides process descriptions, input processes, output processes
3. (What) Entity Relationship Model – identifies and describes the organizational materials and their relationships
4. (Who) Organizational Unit & Role Relationship Model – identifies enterprise roles and units and the relationships between them
5. (Where) Locations Model – identifies enterprise locations and the relationships between them
6. (When) Event Model – identifies and describes events and cycles related by time

Logical

1. (Why) Rules Diagram – identifies and describes rules that apply constraints to processes and entities without regard to physical or technical implementation
2. (How) Process Diagram – identifies and describes process transitions expressed as verb-noun phrases without regard to physical or technical implementation
3. (What) Data Model Diagram – identifies and describes entities and their relationships without regard to physical or technical implementation
4. (Who) Role Relationship Diagram – identifies and describes roles and their relations to other roles by types of deliverables without regard to physical or technical implementation

5. (Where) Locations Diagram – identifies and describes locations used to access, manipulate, and transfer entities and processes without regard to physical or technical implementation
6. (When) Event Diagram – identifies and describes events related to each other in sequence, cycles occur within and between events, without regard to physical or technical implementation

Physical

1. (Why) Rules Specification – expressed in a formal language; consists of rule name and structured logic to specify and test rule state
2. (How) Process Function Specification – expressed in a technology specific language, hierarchical process elements are related by process calls
3. (What) Data Entity Specification – expressed in a technology specific format; each entity is defined by name, description, and attributes; shows relationships
4. (Who) Role Specification – expresses roles performing work and workflow components at the work product detailed specification level
5. (Where) Location Specification – expresses the physical infrastructure components and their connections
6. (When) Event Specification – expresses transformations of event states of interest to the enterprise

Detailed Representation

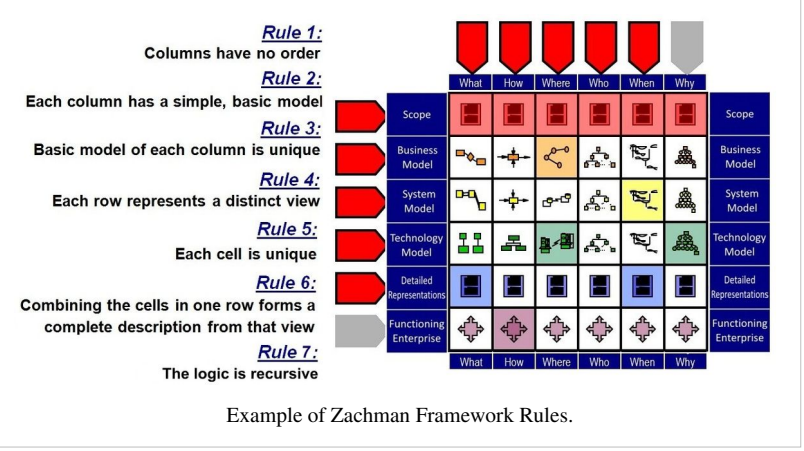
Eventually the cells with the detailed representation give Rules detail for (Why); Process detail for (How); Data detail for (What); Role detail for (Who); Location detail for (Where); and Event detail for (When).

There is a sixth row in the current Zachman framework, but it is not used for enterprise architecture — while the enterprise is described by rows one to six, enterprise architecture uses only rows one to five, thus only five rows are shown here.[3]

Since the product development (i.e., architectural artifact) in each cell or the problem solution embodied by the cell is the answer to a question from a perspective, typically, the models or descriptions are higher-level depictions or the surface answers of the cell. The refined models or designs supporting that answer are the detailed descriptions within the cell. Decomposition (i.e., drill down to greater levels of detail) takes place within each cell. If a cell is not made explicit (defined), it is implicit (undefined). If it is implicit, the risk of making assumptions about these cells exists. If the assumptions are valid, then time and money are saved. If, however, the assumptions are invalid, it is likely to increase costs and exceed the schedule for implementation.[26]

Framework set of rules

The framework comes with a set of rules:[29]



Example of Zachman Framework Rules.

- *Rule 1 The columns have no order* : The columns are interchangeable but cannot be reduced or created
- *Rule 2 Each column has a simple generic model* : Every column can have its own meta-model
- *Rule 3 The basic model of each column must be unique* : The basic model of each column, the relationship objects and the structure of it is unique. Each relationship object is interdependent but the representation objective is unique.
- *Rule 4 Each row describes a distinct, unique perspective* : Each row describes the view of a particular business group and is unique to it. All rows are usually present in most hierarchical organizations.

- *Rule 5 Each cell is unique* : The combination of 2,3 & 4 must produce unique cells where each cell represents a particular case. Example: A2 represents business outputs as they represent what are to be eventually constructed.
- *Rule 6 The composite or integration of all cell models in one row constitutes a complete model from the perspective of that row* : For the same reason as for not adding rows and columns, changing the names may change the fundamental logical structure of the Framework.
- *Rule 7 The logic is recursive* : The logic is relational between two instances of the same entity.

The framework is generic in that it can be used to classify the descriptive representations of any physical object as well as conceptual objects such as enterprises. It is also recursive in that it can be used to analyze the architectural composition of itself. Although the framework will carry the relation from one column to the other, it is still a fundamentally structural representation of the enterprise and not a flow representation.

Flexibility in level of detail

One of the strengths of the Zachman Framework is that it explicitly shows a comprehensive set of views that can be addressed by enterprise architecture.[12] Some feel that following this model completely can lead to too much emphasis on documentation, as artifacts would be needed for every one of the thirty cells in the framework.

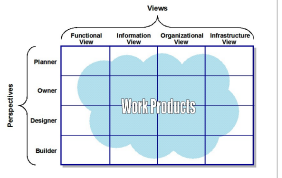
John Zachman clearly states in his documentation, presentations, and seminars that, as framework, there is flexibility in what depth and breadth of detail is required for each cell of the matrix based upon the importance to a given organization. An automaker, whose business goals may necessitate an inventory and process-driven focus, could find it beneficial to focus their documentation efforts on **What** and **How** columns. Whereas a travel agent company, whose business is more concerned with people and event-timing, could find it beneficial to focus their documentation efforts on **Who** and **When** columns. However, there is no escaping the **Why** column's importance as it provides the business drivers for all the other columns.

Applications and influences

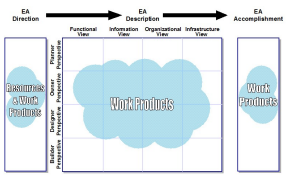
Since the 1990s the Zachman Framework has been widely used as a means of providing structure for Information Engineering-style enterprise modeling.[30] The Zachman Framework can be applied both in commercial companies and in government agencies. Within a government organization the framework can be applied to an entire agency at an abstract level, or it can be applied to various departments, offices, programs, subunits and even to basic operational entities.[31]

Customization

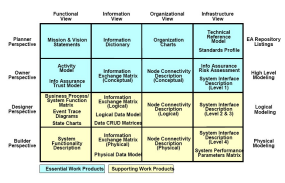
Zachman Framework is applied in customized frameworks such as the TEAF, built around the similar frameworks, the TEAF matrix.



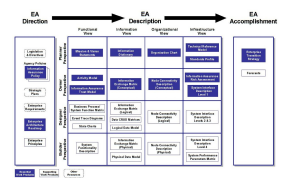
TEAF Matrix of Views and Perspectives.



Framework for EA Direction, Description, and Accomplishment Overview.



TEAF Products.



TEAF Work Products for EA Direction, Description, and Accomplishment.

Other sources:

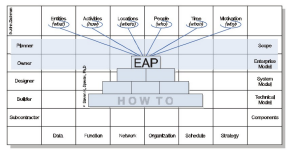
- The TEAF matrix is called a customization sample, see *here* [32], p. 22

Standards based on the Zachman Framework

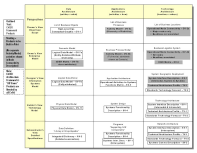
Zachman Framework is also used as a framework to describe standards, for example standards for healthcare and healthcare information system. Each cell of the framework contains such a series of standards for healthcare and healthcare information system.[33]

Mapping other frameworks

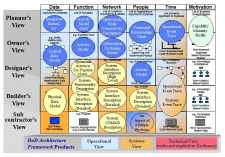
Another application of the Zachman Framework is as reference model for other enterprise architectures, see for example these four:



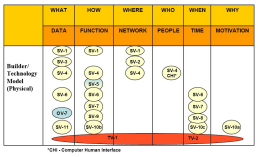
EAP mapped to the Zachman Framework, 1999



Mapping the C4ISR, 1999



DoD Products Map to the Zachman Framework Cells, 2003.



Mapping a part of the DoDAF, 2007.

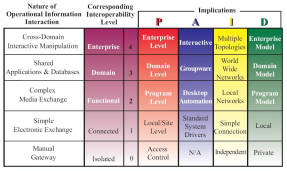
Other examples:

- Analysis of the Rational Unified Process as a Process,[34]
- How the Model-driven architecture (MDA) models used in software development map to the Zachman Framework.[35]
- Mapping the IEC 62264 models onto the Zachman framework for analysing products information traceability.[36]
- Mapping the TOGAF Architecture Development Method (e.g. the methodology) to the Zachman Framework.[6]

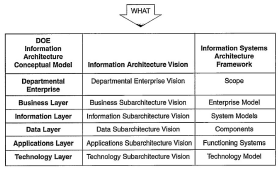
Base for other enterprise architecture frameworks

Less obvious are the ways the original Zachman framework has stimulated the development of other enterprise architecture frameworks, such as in the NIST Enterprise Architecture Model, the C4ISR AE, the DOE AE, and the DoDAF:

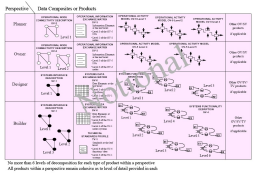
NIST Enterprise Architecture Model.[26]



C4ISR AE, 1997.



DOE AE, 1998.



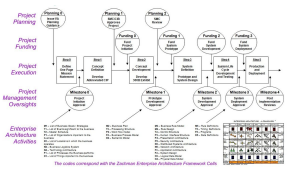
DODAF, 2003.

- The Federal Enterprise Architecture Framework (FEAF) is based on the Zachman Framework but only addresses the first three columns of Zachman, using slightly different names, and focuses in the top of the three rows.[37] (see *here* [38])

Example: One-VA Enterprise Architecture

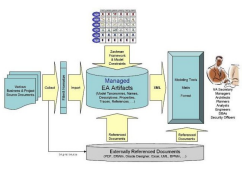
The Zachman Framework methodology has for example been used by the United States Department of Veterans Affairs (VA) to develop and maintain its One-VA Enterprise Architecture in 2001. This methodology required defining all aspects of the VA enterprise from a business process, data, technical, location, personnel, and requirements perspective. The next step in implementing the methodology has been to define all functions related to each business process and identify associated data elements. Once identified, duplication of function and

inconsistency in data definition can be identified and resolved, .[39]



Integrated Process Flow for VA IT Projects (2001)

VA Zachman Framework Portal



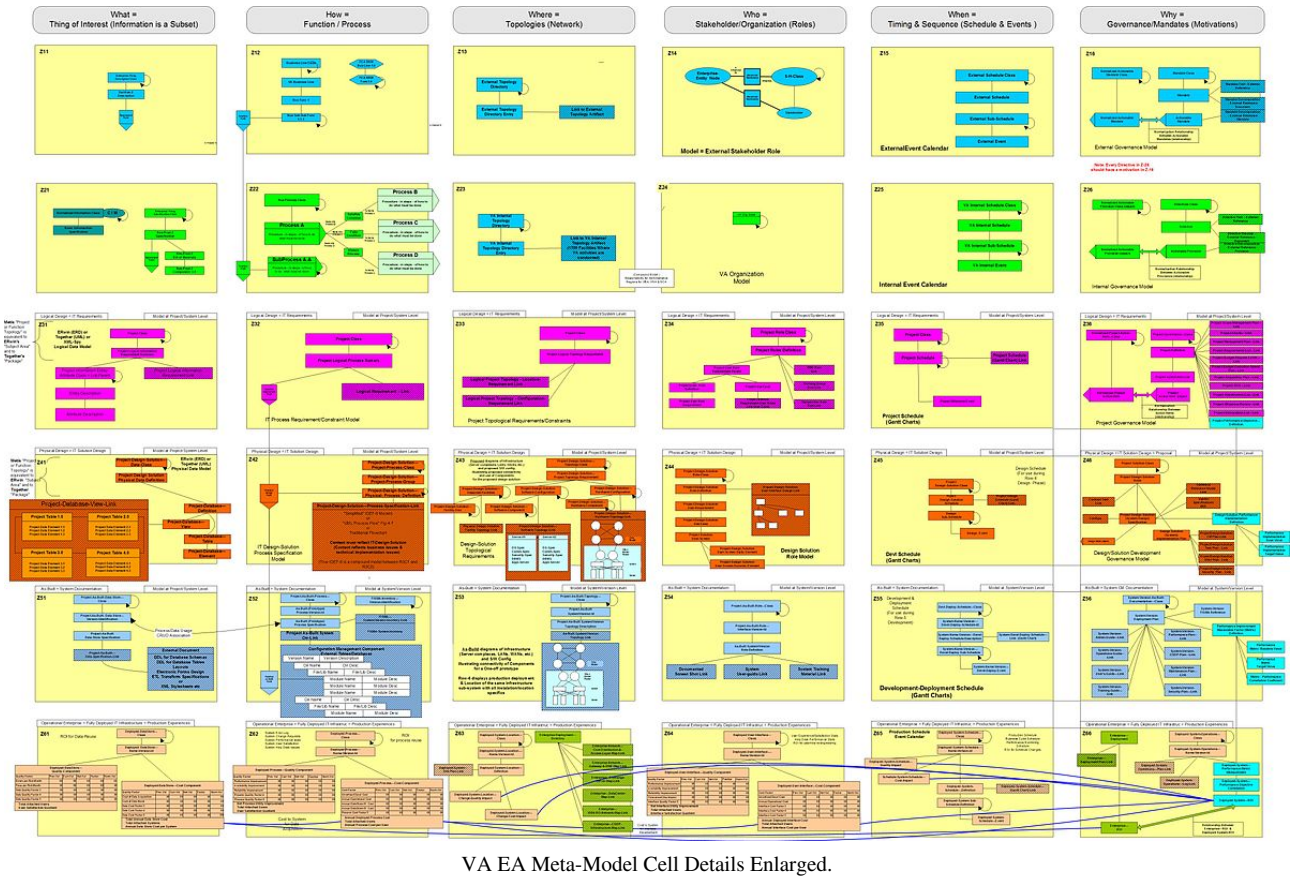
VA EA Repository Introduction (2008)

A Tutorial on the Zachman Architecture Framework

The Department of Veterans Affairs at the beginning of the 21st century planned to implement an enterprise architecture fully based on the Zachman Framework.

- The Zachman Framework was used as a reference model to initiate enterprise architecture planning in 2001.
- Somewhere in between the VA Zachman Framework Portal was constructed.
- This VA Zachman Framework Portal is still in use as a reference model for example in the determination of EA information collected from various business and project source documents.
- Now somewhere in the past this "A Tutorial on the Zachman Architecture Framework".

Eventually an enterprise architecture repository was created at the macro level by the Zachman framework and at a cell level by the meta-model outlined below.[40]



VA EA Meta-Model Cell Details Enlarged.

This diagram[41] has been incorporated within the VA-EA to provide a symbolic representation of the metamodel it used, to describe the One-VA Enterprise Architecture and to build an EA Repository without the use of Commercial EA Repository Software. It was developed using an object oriented database within the Caliber-RM Software Product. Caliber-RM is intended to be used as a software configuration management tool; not as an EA repository.

However, this tool permitted defining entities and relationships and for defining properties upon both entities and relationships, which made it sufficient for building an EA repository, considering the technology available in early 2003. The personal motivation in selecting this tool was that none of the commercial repository tools then available provided a true Zachman Framework representation, and were highly proprietary, making it difficult to incorporate components from other vendors or from open source.

This diagram emphasizes several important interpretations of the Zachman Framework and its adaptation to information technology investment management.

1. Progressing through the rows from top to bottom, one can trace-out the Systems Development Life Cycle (SDLC) which is a de facto standard across the Information Industry;
2. The diagram emphasizes the importance of the often-neglected Zachman Row-Six (the Integrated, Operational Enterprise View). Representations in Mr. Zuech's interpretation of Zachman row-six consist, largely, of measurable service improvements and cost savings/avoidance that result from the business process and technology innovations that were developed across rows two through five.

Row-six provides measured return on investment for Individual Projects and, potentially, for the entire investment portfolio. Without row-six the Framework only identifies sunk-cost, but the row-six ROI permits it to measure benefits and to be used in a continuous improvement process, capturing best practices and applying them back through row-two.

References

[1] "John Zachman's Concise Definition of the The Zachman Framework" (http://zachman.com/about-the-zachman-framework). Zachman International. 2008. .

[2] "The Zachman Framework: The Official Concise Definition" (http://zachman.com/about-the-zachman-framework). Zachman International. 2008. .

[3] "The Zachman Framework Evolution" (http://zachman.com/ea-articles-reference/54-the-zachman-framework-evolution). Zachman International. April, 2009. .

[4] *A Comparison of the Top Four Enterprise Architecture Methodologies* (http://msdn2.microsoft.com/en-us/library/bb466232.aspx), Roger Sessions, Microsoft Developer Network Architecture Center,

[5] "A framework for information systems architecture" (http://zachman.com/images/ZI_PIcs/ibmsj2603e.pdf). IBM SYSTEMS JOURNAL, VOL 26. NO 3,. 1987. .

[6] The Open Group (1999–2006). "ADM and the Zachman Framework" (http://www.theopengroup.org/architecture/togaf8-doc/arch/chap39.html) in: *TOGAF 8.1.1 Online*. Accessed 25 Jan 2009.

[7] William H. Inmon, John A. Zachman, Jonathan G. Geiger (1997). *Data Stores, Data Warehousing, and the Zachman Framework: Managing Enterprise Knowledge*. McGraw-Hill, 1997. ISBN 0-07-031429-2.

[8] Pete Sawyer, Barbara Paech, Patrick Heymans (2007). *Requirements Engineering: Foundation for Software Quality*. page 191.

[9] Kathleen B. Hass (2007). *The Business Analyst as Strategist: Translating Business Strategies Into Valuable Solutions*. page 58.

[10] Harold F. Tipton, Micki Krause (2008). *Information Security Management Handbook, Sixth Edition, Volume 2*. page 263.

[11] O'Rourke, Fishman, Selkow (2003). *Enterprise Architecture Using the Zachman Framework*. page 9.

[12] James McGovern et al. (2003). *A Practical Guide to Enterprise Architecture*. p. 127-129.

[13] Marc Lankhorst et al. (2005). *Enterprise Architecture at Work*. p. 24.

[14] "Business Systems Planning and Business Information Control Study: A comparisment (http://www.research.ibm.com/journal/sj/211/ibmsj2101D.pdf). In: *IBM Systems Journal*, vol 21, no 3, 1982. p. 31-53.

[15] John A. Zachman (1987). " A Framework for Information Systems Architecture" (http://www.research.ibm.com/journal/50th/applications/zachman.html). In: *IBM Systems Journal*, vol 26, no 3. IBM Publication G321-5298.

[16] Durward P. Jackson (1992). "Process-Based Planning in Information Resource Management". In: *Emerging Information Technologies for Competitive Advantage and Economic Development*. Proceedings of 1992 Information Resources Management Association International Conference. *Mehdi Khosrowpour (ed). ISBN 1-878289-17-9.*

[17] Alain Wegmann et al. (2008). "Augmenting the Zachman Enterprise Architecture Framework with a Systemic Conceptualization" (http://infoscience.epfl.ch/record/129325/files/Wegmann_et_al-SEAM_&_Zachman-EDOC2008.pdf). Presented at the 12th IEEE International EDOC Conference (EDOC 2008), München, Germany, September 15–19, 2008.

[18] John F. Sowa and John Zachman (1992). "Extending and Formalizing the Framework for Information Systems Architecture" (http://www.research.ibm.com/journal/sj/313/sowa.pdf) In: *IBM Systems Journal*, Vol 31, no.3, 1992. p. 590-616.

[19] Stan Locke (2008). "Enterprise Convergence in Our Lifetime" (http://www.ies.aust.com/ten/TEN42.htm#Enterprise_Convergence) In: THE ENTERPRISE NEWSLETTER, TEN42 September 16, 2008

[20] John A. Zachman (1997). " Concepts of the Framework for Enterprise Architecture: Background, Description and Utility (http://www.ies.aust.com/PDF-papers/zachman3.pdf)". Zachman International. Accessed 19 Jan 2009.
[21] R.W. Matthews. &. W.C. McGee (1990). "Data Modeling for Software Development" (http://www.research.ibm.com/journal/sj/292/ibmsj2902F.pdf). in: *IBM Systems Journal" 29(2). pp. 228–234*
[22] Jaap Schekkerman (2003). *How to Survive in the Jungle of Enterprise Architecture Frameworks*. page 139-144.
[23] Vladan Jovanovic, Stevan Mrdalj & Adrian Gardiner (2006). A Zachman Cube (http://www.iacis.org/iis/2006_iis/PDFs/Jovanovic_Mrdalj_Gardiner.pdf). In: *Issues in Information Systems*. Vol VII, No. 2, 2006 p. 257-262.
[24] VA Enterprise Architecture Innovation Team (2001). *Enterprise Architecture: Strategy, Governance, & Implementation* (http://www.va.gov/oirm/architecture/ea/2002/VAEAVersion-10-01.pdf) report Department of Veterans Affairs, August, 2001.
[25] The government information factory and the Zachman Framework (http://www.inmongif.com/_fileCabinet/gifzach.pdf) by W. H. Inmon, 2003. p. 4. Accessed July 14, 2009.
[26] The Chief Information Officers Council (1999). Federal Enterprise Architecture Framework Version 1.1 (http://www.cio.gov/documents/fedarch1.pdf). September 1999
[27] US Department of Veterans Affairs (2002) A Tutorial on the Zachman Architecture Framework (http://www.va.gov/oirm/architecture/EA/theory/tutorial.ppt). Accessed 06 Dec 2008.
[28] Bill Inmon called this image "A simple example of The Zachman Framework" in the article John Zachman - One of the Best Architects I Know (http://www.b-eye-network.in/print/1962) Originally published 17 November 2005.
[29] Adapted from: Sowa, J.F. & J.A. Zachman, 1992, and Inmon, W.H, J.A. Zachman, & J.G. Geiger, 1997. University of Omaha (http://www.isqa.unomaha.edu/vanvliet/arch/ISA/isa.htm)
[30] Ian Graham (1995). *Migrating to Object Technology: the semantic object modelling approach*. Addison-Wesley, ISBN 0-201-59389-0. p. 322.
[31] Jay D. White (2007). *Managing Information in the Public Sector*. p. 254.
[32] http://www.mega.com/wp/active/document/company/wp_mega_zachman_en.pdf
[33] ZACHMAN ISA FRAMEWORK FOR HEALTHCARE INFORMATICS STANDARDS (http://apps.adcom.uci.edu/EnterpriseArch/Zachman/Resources/ExampleHealthCareZachman.pdf), 1997.
[34] DJ de Villiers (2001). "Using the Zachman Framework to Assess the Rational Unified Process" (http://www.ibm.com/developerworks/rational/library/content/RationalEdge/mar01/UsingtheZachmanFrameworktoAssesstheRUPMar01.pdf), In: *The Rational Edge* Rational Software 2001.
[35] David S. Frankel et al. (2003) *The Zachman Framework and the OMG's Model Driven Architecture* (http://www.bptrends.com/publicationfiles/09-03 WP Mapping MDA to Zachman Framework.pdf) White paper. Business Process Trends.
[36] Hervé Panetto, Salah Baïna, Gérard Morel (2007). Mapping the models onto the Zachman framework for analysing products information traceability : A case Study (http://hal.archives-ouvertes.fr/docs/00/11/91/96/PDF/Panetto_et_al_JIM.pdf).
[37] Roland Traunmüller (2004). *Electronic Government* p. 51
[38] http://books.google.nl/books?id=QjB5c_v-uMwC&pg=PA51&dq=%22Zachman+Framework%22+updated&lr=lang_en&as_brr=0&as_pt=ALLTYPES
[39] Statement of Dr. John A. Gauss, Assistant Secretary for Information and Technology, Department of Veterans Affairs (http://www.va.gov/oca/testimony/hvac/soi/13mr02it.asp), before the Subcommittee on Oversight and Investigations Committee on Veterans' Affairs U.S. House of Representatives. March 13, 2002.
[40] Meta-Model Cell Details (http://www.va.gov/oit/ea/4_3/process/modeling/metamodel.html) Accessed 25 Dec 2009
[41] This diagram is the exclusive work of Albin Martin Zuech of Annapolis Maryland, who placed it in the public domain in 2001. Al Zuech maintains the original visio diagram in numerous stages of its development between 2000 and present. Al Zuech was the Director, Enterprise Architecture Service at the Department of Veterans Affairs from 2001 until 2007.

External links

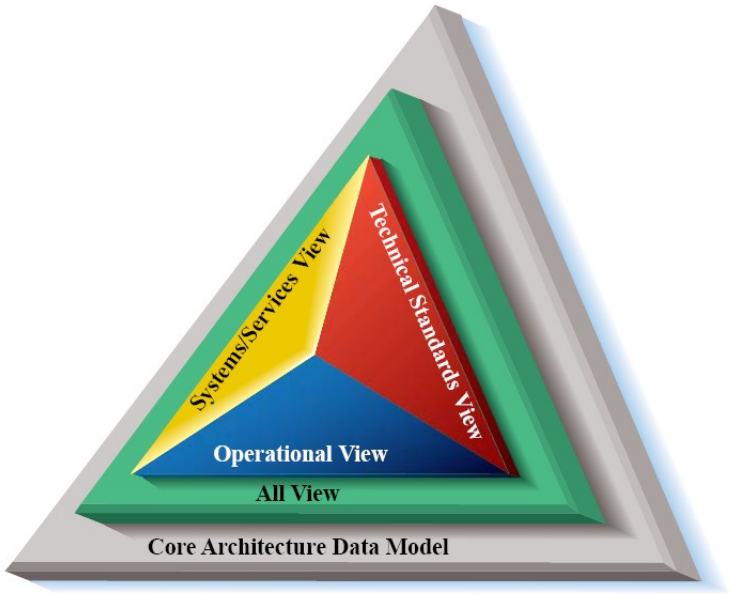
- The Zachman Framework: The Official Concise Definition (http://zachman.com/about-the-zachman-framework) by John A. Zachman at Zachman International, 2009.
- The Zachman Framework Evolution (http://zachman.com/ea-articles-reference/54-the-zachman-framework-evolution): overview of the evolution of the Zachman Framework by John P. Zachman at Zachman International, April 2009.
- UML, RUP, and the Zachman Framework: Better together (http://www.ibm.com/developerworks/rational/library/nov06/temnenco/), by Vitalie Temnenco, IBM, 15 Nov 2006.

Defense industry frameworks

Department of Defense Architecture Framework

The **Department of Defense Architecture Framework (DoDAF)** is an architecture framework for the United States Department of Defense, that provides structure for a specific stakeholder concern through viewpoints organized by various views.

DoDAF defines a set of views that act as mechanisms for visualizing, understanding, and assimilating the broad scope and complexities of an architecture description through tabular, structural, behavioral, ontological, pictorial, temporal or graphical means.



DoD Architecture Framework.[1]

It is especially suited to large systems with complex integration and interoperability challenges, and is apparently unique in its use of "operational views" detailing the external customer's operating domain in which the developing system will operate.[2]

Overview

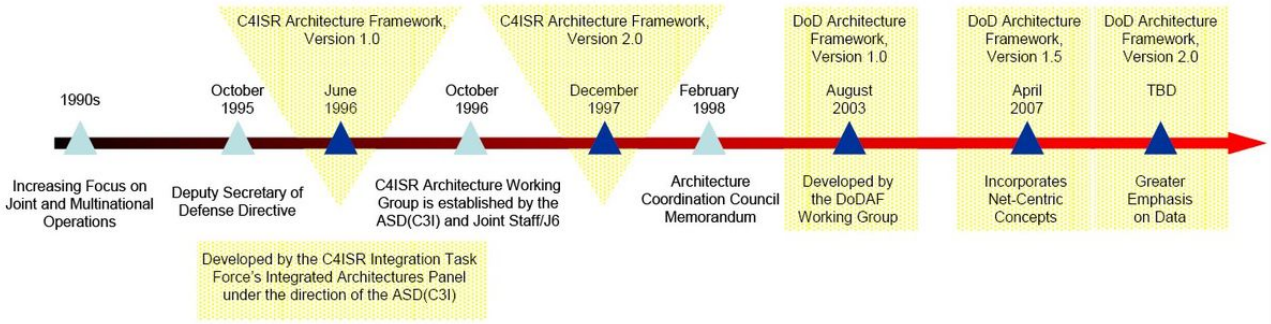
The Department of Defense Architecture Framework (DoDAF) provides a foundational framework for developing and representing architecture descriptions that ensure a common denominator for understanding, comparing, and integrating architectures across organizational, Joint, and multinational boundaries. It establishes data element definitions, rules, and relationships and a baseline set of products for consistent development of systems, integrated, or federated architectures. These architecture descriptions may include Families of Systems (FoS), Systems of Systems (SoS), and net-centric capabilities for interoperating and interacting in the NCE.[1]

All major U.S. Government Department of Defense (DoD) weapons and information technology system acquisitions are required to develop and document an Enterprise Architecture (EA) using the views prescribed in the DoDAF. While it is clearly aimed at military systems, DoDAF has broad applicability across the private, public and voluntary sectors around the world, and represents one of a large number of systems architecture frameworks.[3]

- The purpose of DoDAF is to define concepts and models usable in DoD's six core processes:[4]
  1. Capabilities Integration and Development (JCIDS)
  2. Planning, Programming, Budgeting, and Execution (PPBE)
  3. Acquisition System (DAS)
  4. Systems Engineering (SE)
  5. Operations Planning
  6. Capabilities Portfolio Management (CPM)

- In addition, DoDAF 2.0's specific goals were to:[4]
  1. Establish guidance for architecture content as a function of purpose – “fit for purpose”
  2. Increase utility and effectiveness of architectures via a rigorous data model – the DoDAF Meta Model (DM2) -- so the architectures can be integrated, analyzed, and evaluated to mathematical precision.

History



Evolution of the DoDAF Framework since the 1990s.[1]

Note, where the diagram states *TBD*, the DoDAF V2.0 was promulgated on May 28, 2009. The first version of the development DoDAF was developed in the 1990s under the name C4ISR architectural Architecture Framework. C4ISR stand for The Command, Control, Communications, Computers, and Intelligence, Surveillance, and Reconnaissance. In the same period the reference model TAFIM, which was initiated in 1986, was further developed. The first C4ISR Architecture Framework v1.0, released 7 June 1996, was created in response to the passage of the Clinger-Cohen Act. It addressed the 1995 Deputy Secretary of Defense directive that a DoD-wide effort be undertaken to define and develop a better means and process for ensuring that C4ISR capabilities were interoperable and met the needs of the warfighter. Continued development effort resulted in December 1997 in the second version, C4ISR Architecture Framework v2.0.[1]

In August 2003 the DoDAF v1.0 was released, which restructured the C4ISR Framework v2.0 to offer guidance, product descriptions, and supplementary information in two volumes and a Desk Book. It broadened the applicability of architecture tenets and practices to all Mission Areas rather than just the C4ISR community. This document addressed usage, integrated architectures, DoD and Federal policies, value of architectures, architecture measures, DoD decision support processes, development techniques, analytical techniques, and the CADM v1.01, and moved towards a repository-based approach by placing emphasis on architecture data elements that comprise architecture products.[1]

In February 2004 the documentation of Version 1.0 was released with volume "I: Definitions and Guidelines", "II: Product Descriptions" and a "Deskbook". In April 2007 the Version 1.5 was released with a documentation of "Definitions and Guidelines", "Product Descriptions" and "Architecture Data Description".[5]

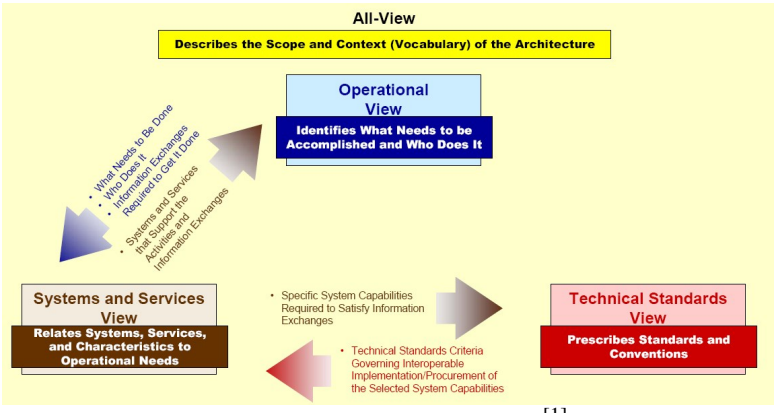
On May 28, 2009 DoDAF v2.0 was approved by the Department of Defense.[6] DoDAF V2.0 is published on a public website.[7]

Other derivative frameworks based on DoDAF include the NATO Architecture Framework (NAF) and Ministry of Defence (United Kingdom) Architecture Framework (MODAF). Like other EA approaches, for example The Open Group Architecture Framework (TOGAF), DoDAF is organized around a shared repository to hold work products. The repository is defined by the Core Architecture Data Model 2.0 (CADM -- essentially a common database schema) and the DoD Architecture Registry System (DARS). A key feature of DoDAF is interoperability, which is organized as a series of levels, called Levels of Information System Interoperability (LISI). The developing system must not only meet its internal data needs but also those of the operational framework into which it is set.

DoDAF V1.5 Views

The DoDAF V1.5 defines a set of products, a view model, that act as mechanisms for visualizing, understanding, and assimilating the broad scope and complexities of an architecture description through graphic, tabular, or textual means. These products are organized under four views:

- overarching All View (AV)
- Operational View (OV)
- Systems View (SV)
- Technical Standards View (TV)



DoDAF V1.5 Linkages Among Views.[1]

Each view depicts certain perspectives of an architecture as described below. Only a subset of the full DoDAF viewset is usually created for each system development. The figure represents the information that links the operational view, systems and services view, and technical standards view. The three views and their interrelationships – driven by common architecture data elements – provide the basis for deriving measures such as interoperability or performance, and for measuring the impact of the values of these metrics on operational mission and task effectiveness.[1]

All View (AV)

AV products provide overarching descriptions of the entire architecture and define the scope and context of the architecture. The DoDAF V1.5 AV products are defined as:

AV-1 Overview and Summary Information

> Scope, purpose, intended users, environment depicted, analytical findings (if applicable)

AV-2 Integrated Dictionary

> Definitions of all terms used in all products.

Operational View (OV)

Operational View (OV) products provide descriptions of the tasks and activities, operational elements, and information exchanges required to accomplish DoD missions. The OV provides textual and graphical representations of operational nodes and elements, assigned tasks and activities, and information flows between nodes. It defines the type of information exchanged, the frequency of exchanges, the tasks and activities supported by these exchanges and the nature of the exchanges. The DoDAF V1.5 OV products are defined as:

OV-1 High Level Operational Concept Graphic

> High level graphical and textual description of operational concept (high level organizations, missions, geographic configuration, connectivity, etc.).

OV-2 Operational Node Connectivity Description

> Operational nodes, activities performed at each node, and connectivities and information flow between nodes.

OV-3 Operational Information Exchange Matrix

> Information exchanged between nodes and the relevant attributes of that exchange such as media, quality, quantity, and the level of interoperability required.

OV-4 Organizational Relationships Chart

Command, control, coordination, and other relationships among organizations.

OV-5 Operational Activity Model

Activities, relationships among activities, inputs and outputs. In addition, overlays can show cost, performing nodes, or other pertinent information.

OV-6a Operational Rules Model

One of the three products used to describe operational activity sequence and timing that identifies the business rules that constrain the operation.

OV-6b Operational State Transition Description

One of the three products used to describe operational activity sequence and timing that identifies responses of a business process to events.

OV-6c Operational Event-Trace Description

One of the three products used to describe operational activity sequence and timing that traces the actions in a scenario or critical sequence of events.

OV-7 Logical Data Model

Documentation of the data requirements and structural business process rules of the Operational View. (In DoDAF V1.5. This corresponds to DIV-2 in DoDAF V2.0.)

Systems and Services View (SV)

SV is a set of graphical and textual products that describe systems and services and interconnections providing for, or supporting, DoD functions. SV products focus on specific physical systems with specific physical (geographical) locations. The relationship between architecture data elements across the SV to the OV can be exemplified as systems are procured and fielded to support organizations and their operations. The DoDAF V1.5 SV products are:

SV-1 Systems/Services Interface Description

Depicts systems nodes and the systems resident at these nodes to support organizations/human roles represented by operational nodes of the OV-2. SV-1 also identifies the interfaces between systems and systems nodes.

SV-2 Systems/Services Communications Description

Depicts pertinent information about communications systems, communications links, and communications networks. SV-2 documents the kinds of communications media that support the systems and implements their interfaces as described in SV-1. Thus, SV-2 shows the communications details of SV-1 interfaces that automate aspects of the needlines represented in OV-2.

SV-3 Systems-Systems, Services-Systems, Services-Services Matrices

provides detail on the interface characteristics described in SV-1 for the architecture, arranged in matrix form.

SV-4a/SV-4b Systems/Services Functionality Description

The SV-4a documents system functional hierarchies and system functions, and the system data flows between them. The SV-4 from DoDAF v1.0 is designated as 'SV-4a' in DoDAF v1.5. Although there is a correlation between OV-5 or business-process hierarchies and the system functional hierarchy of SV-4a, it need not be a one-to-one mapping, hence, the need for the Operational Activity to Systems Function Traceability Matrix (SV-5a), which provides that mapping.

SV-5a, SV-5b, SV-5c Operational Activity to Systems Function, Operational Activity to Systems and Services Traceability Matrices

Operational Activity to SV-5a and SV-5b is a specification of the relationships between the set of operational activities applicable to an architecture and the set of system functions applicable to that architecture. The SV-5 and extension to the SV-5 from DoDAF v1.0 is designated as 'SV-5a' and 'SV-5b' in DoDAF v1.5 respectively.

SV-6 Systems/Services Data Exchange Matrix

Specifies the characteristics of the system data exchanged between systems. This product focuses on automated information exchanges (from OV-3) that are implemented in systems. Non-automated information exchanges, such as verbal orders, are captured in the OV products only.

SV-7 Systems/Services Performance Parameters Matrix

Specifies the quantitative characteristics of systems and system hardware/software items, their interfaces (system data carried by the interface as well as communications link details that implement the interface), and their functions. It specifies the current performance parameters of each system, interface, or system function, and the expected or required performance parameters at specified times in the future. Performance parameters include all technical performance characteristics of systems for which requirements can be developed and specification defined. The complete set of performance parameters may not be known at the early stages of architecture definition, so it should be expected that this product will be updated throughout the system's specification, design, development, testing, and possibly even its deployment and operations life-cycle phases.

SV-8 Systems/Services Evolution Description

Captures evolution plans that describe how the system, or the architecture in which the system is embedded, will evolve over a lengthy period of time. Generally, the timeline milestones are critical for a successful understanding of the evolution timeline.

SV-9 Systems/Services Technology Forecast

Defines the underlying current and expected supporting technologies that have been targeted using standard forecasting methods. Expected supporting technologies are those that can be reasonably forecast given the current state of technology and expected improvements. New technologies should be tied to specific time periods, which can correlate against the time periods used in SV-8 milestones.

SV-10a Systems/Services Rules Model

Describes the rules under which the architecture or its systems behave under specified conditions.

SV-10b Systems/Services State Transition Description

A graphical method of describing a system (or system function) response to various events by changing its state. The diagram basically represents the sets of events to which the systems in the architecture will respond (by taking an action to move to a new state) as a function of its current state. Each transition specifies an event and an action.

SV-10c Systems/Services Event-Trace Description

Provides a time-ordered examination of the system data elements exchanged between participating systems (external and internal), system functions, or human roles as a result of a particular scenario. Each event-trace diagram should have an accompanying description that defines the particular scenario or situation. SV-10c in the Systems and Services View may reflect system-specific aspects or refinements of critical sequences of events described in the Operational View.

SV-11 Physical Schema

One of the architecture products closest to actual system design in the Framework. The product defines the structure of the various kinds of system data that are utilized by the systems in the architecture. (In DoDAF V1.5. This corresponds to DIV-3 in DoDAF V2.0.)

Technical Standards View (TV)

TV products define technical standards, implementation conventions, business rules and criteria that govern the architecture. The DoDAF V1.5 TV products are as follows:

- **TV-1 Technical Standards Profile** - Extraction of standards that applies to the given architecture. (In DoDAF V1.5. Renamed to StdV-1 in DoDAF V2.0.)
- **TV-2 Technical Standards Forecast** - Description of emerging standards that are expected to apply to the given architecture, within an appropriate set of timeframes. (In DoDAF V1.5. Renamed to StdV-2 in DoDAF V2.0.)

DoDAF V2.0 Viewpoints

In DoDAF V2.0 architectural viewpoints are composed of data that has been organized to facilitate understanding. To align with ISO Standards, where appropriate, the terminology has changed from Views to Viewpoint (e.g., the Operational View is now the Operational Viewpoint).

- **All Viewpoint (AV)** - Describes the overarching aspects of architecture context that relate to all viewpoints.
- **Capability Viewpoint (CV) (New in DoDAF V2.0)** - Articulates the capability requirements, the delivery timing, and the deployed capability.
- **Data and Information Viewpoint (DIV) (New in DoDAF V2.0)** - Articulates the data relationships and alignment structures in the architecture content for the capability and operational requirements, system engineering processes, and systems and services.

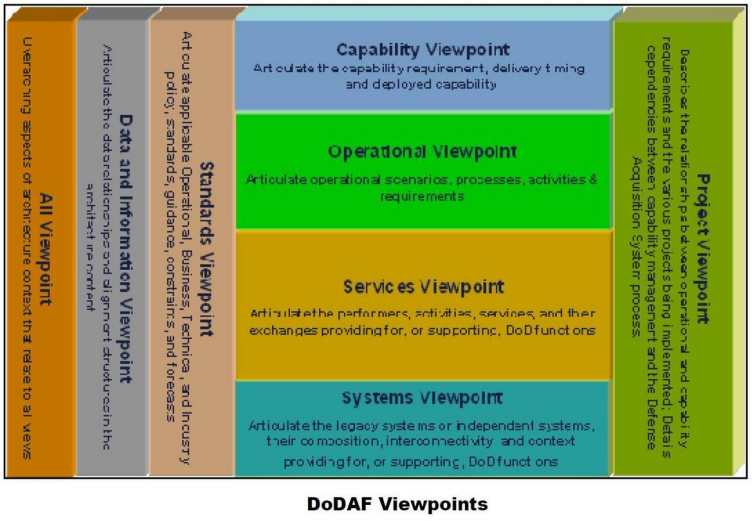


Diagram of DoDAF V2.0 Viewpoints.[8]

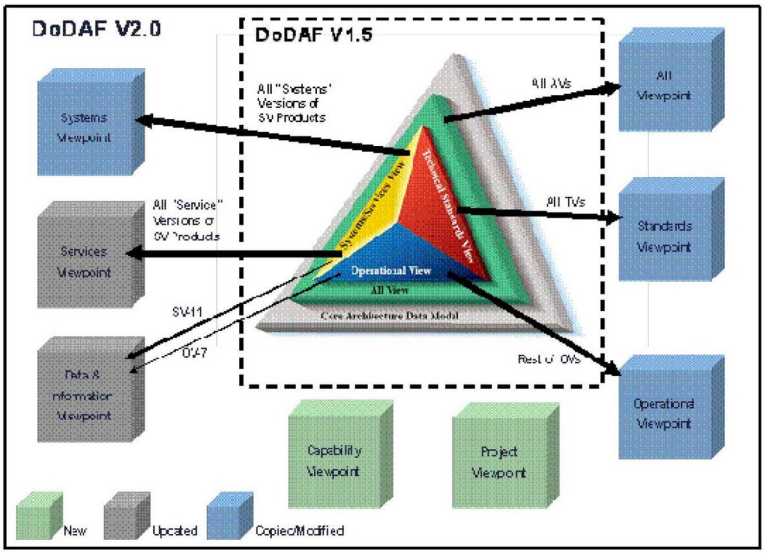
- **Operational Viewpoint (OV)** - Includes the operational scenarios, activities, and requirements that support capabilities.
- **Project Viewpoint (PV) (New in DoDAF V2.0)** - Describes the relationships between operational and capability requirements and the various projects being implemented. The Project Viewpoint also details dependencies among capability and operational requirements, system engineering processes, systems design, and services design within the Defense Acquisition System process.
- **Services Viewpoint (SvcV) (New in DoDAF V2.0)** - Presents the design for solutions articulating the Performers, Activities, Services, and their Exchanges, providing for or supporting operational and capability functions.
- **Standards Viewpoint (StdV) (Renamed from Technical Standards View TV)** - Articulates the applicable operational, business, technical, and industry policies, standards, guidance, constraints, and forecasts that apply to capability and operational requirements, system engineering processes, and systems and services.
- **Systems Viewpoint (SV)** - Articulates, for Legacy support, the design for solutions articulating the systems, their composition, interconnectivity, and context providing for or supporting operational and capability functions.

Note, *System* has changed in DoDAF V2.0 from DoDAF V1.5: System is not just computer hardware and computer software. System is now defined in the general sense of an assemblage of components - machine, human - that perform activities (since they are subtypes of Performer) and are interacting or interdependent. This could be

anything, i.e., anything from small pieces of equipment that have interacting or interdependent elements, to Family of Systems (FoS) and System of Systems (SoS). Note that Systems are made up of Materiel (e.g., equipment, aircraft, and vessels) and Personnel Types.

Relationship of DoDAF V1.5 Views with DoDAF V2.0 Viewpoints

The first figure shows the overall evolution of DoDAF V1.5 Views to DoDAF V2.0 Viewpoints.



DoDAF V1.5 Evolution to DoDAF V2.0

Evolution of DoDAF V1.5 Views to DoDAF V2.0 Viewpoints.[9]

The second figure shows the specific mapping of individual DoDAF V1.5 Views to the corresponding DoDAF V2.0 Viewpoints.

DoDAF V1.5 Support. The architectures for DoDAF V1.0 and DoDAF V1.5 may continue to be used. When appropriate (usually indicated by policy or by the decision-maker), DoDAF V1.0 and V1.5 architectures will need to update their architecture. When pre-DoDAF V2.0 architecture is compared with DoDAF V2.0 architecture, concept differences (such as Node) must be defined or explained for the newer architecture.

In regard to DoDAF V1.5 products, they have been transformed into parts of the DoDAF V2.0 models. In most cases, the DoDAF V2.0 Meta-model supports the DoDAF V1.5 data concepts, with one notable exception: Node. Node is a complex, logical concept that is represented with more concrete concepts. The table below indicates the mapping of DoDAF V1.5 products to DoDAF V2.0 models.

DoDAF V2.0 / DoDAF V1.5	Operational Viewpoint	Systems Viewpoint	Services Viewpoint	All Viewpoint	Standards Viewpoint	Data & Information Viewpoint
AV-1				AV-1		
AV-2				AV-2		
OV-1	OV-1					
OV-2	OV-2					
OV-3	OV-3					
OV-4	OV-4					
OV-5	OV-5a, OV-5b					
OV-6a	OV-6a					
OV-6b	OV-6b					
OV-6c	OV-6c					
OV-7						DIV-2
SV-1		SV-1	SvcV-1			
SV-2		SV-2	SvcV-2			
SV-3		SV-3	SvcV-3a, SvcV-3b			
SV-4a		SV-4				
SV-4b			SvcV-4			
SV-5a		SV-5a				
SV-5b		SV-5b				
SV-5c			SvcV-5			
SV-6		SV-6	SvcV-6			
SV-7		SV-7	SvcV-7			
SV-8		SV-8	SvcV-8			
SV-9		SV-9	SvcV-9			
SV-10a		SV-10a	SvcV-10a			
SV-10b		SV-10b	SvcV-10b			
SV-10c		SV-10c	SvcV-10c			
SV-11						DIV-3
TV-1					StdV-1	
TV-2					StdV-2	

Mapping of DoDAF V1.5 Views to DoDAF V2.0 Viewpoints.[10]

Creating an integrated architecture using DoDAF

There are many different approaches for creating an integrated architecture using DoDAF, and for determining which products are required. The approach depends on the requirements and the expected results; i.e., what the resulting architecture will be used for. As one example, the DoDAF v1.0 listed the following products as the "minimum set of products required to satisfy the definition of an OV, SV and TV." One note: while the DoDAF does not list the OV-1 artifact as a core product, its development is strongly encouraged. The sequence of the artifacts listed below gives a suggested order in which the artifacts could be developed. The actual sequence of view generation and their potential customization is a function of the application domain and the specific needs of the effort.

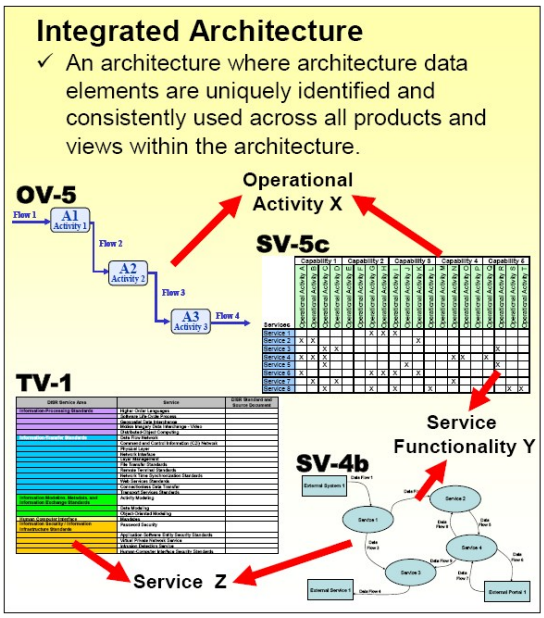


Illustration of the Integrated Architecture.[1]

- AV-1 : Overview and Summary Information
- AV-2 : Integrated Dictionary
- OV-1 : High Level Operational Concept Graphic

- OV-5 : Operational Activity Model
- OV-2 : Operational Node Connectivity Description
- OV-3 : Operational Informational Exchange Matrix
- SV-1 : System Interface Description
- TV-1 : Technical Standards Profile

One concern about the DoDAF is how well these products meet actual stakeholder concerns for any given system of interest. One can view DoDAF products, or at least the 3 views, as ANSI/IEEE 1471-2000 or ISO/IEC 42010 *viewpoints*. But to build an architecture description that corresponds to ANSI/IEEE 1471-2000 or ISO/IEC 42010, it is necessary to clearly identify the stakeholders and their concerns that map to each selected DoDAF product. Otherwise there is the risk of producing products with no customers.

The figure "DoDAF V1.5 Products Matrix" shows how the DoD Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01E specifies which DoDAF V1.5 products are required for each type of analysis, in the context of the Net-Ready Key Performance Parameter (NR-KPP):

Document	Supportability Compliance	DOD Enterprise Architecture Products (IAW DODAF) (see Note 5)																	Data/Service Exposure Sheets	IA Compliance	GTG Compliance
		AV-1 /AV-2	OV-1	OV-2	OV-3	OV-4	OV-5	OV-6C	OV-7	SV-1	SV-2	SV-4	SV-5	SV-6	SV-11	TV-1	TV-2				
ICD			X																		
CDD	X	3	X	X	X	X	X	X			X	X	X	X		2	2	1	X	X	
CPD	X	3	X	X	X	X	X	X	1		X	X	X	X	1	2	2	1	X	X	
ISP	X	3	X	X	X	X	X	X	4		X	X	X	X	4	2	2	1	X	X	
TISP	X	3	X		X		X	X		X			X	X		2	2	1	X	X	
ISP Annex (Svcs/ Apps)	X	3	X				X				X	X	X	X		2	2	1	X	X	
X	Required (PM needs to check with their Component for any additional architectural/regulatory requirements for CDDs, CPDs, ISPs/TISPs. (e.g., HQDA requires the SV-10c)																				
Note 1	Required only when IT and NSS collects, processes, or uses any shared data or when IT and NSS exposes, consumes or implements shared services,																				
Note 2	The TV-1 and TV-2 are built using the DISRonline and must be posted for compliance.																				
Note 3	The AV-1 must be uploaded onto DARS and must be registered in DARS for compliance																				
Note 4	Only required for Milestone C, if applicable (see Note 1)																				
Note 5	The naming of the architecture views is expected to change with the release of DODAF v2.0 (e.g., StdV, SvcV, StdV, DIV). The requirements of this matrix will not change.																				

Table E-1. NR-KPP Products Matrix

DoDAF V1.5 Products Matrix [11]

- Initial Capabilities Document (ICD). Documents the need for a materiel solution to a specific capability gap derived from an initial analysis of alternatives executed by the operational user and, as required, an independent analysis of alternatives. It defines the capability gap in terms of the functional area, the relevant range of military operations, desired effects, and time.
- Capability Development Document (CDD). A document that captures the information necessary to develop a proposed program(s), normally using an evolutionary acquisition strategy. The CDD outlines an affordable increment of militarily useful, logistically supportable and technically mature capability.
- Capability Production Document (CPD). A document that addresses the production elements specific to a single increment of an acquisition program.
- Information Support Plan (ISP). The identification and documentation of information needs, infrastructure support, IT and NSS interface requirements and dependencies focusing on net-centric, interoperability, supportability and sufficiency concerns (DODI 4630.8).
- Tailored Information Support Plan (TISP). The purpose of the TISP process is to provide a dynamic and efficient vehicle for certain programs (ACAT II and below) to produce requirements necessary for I&S Certification. Select program managers may request to tailor the content of their ISP (ref ss). For programs not designated OSD special interest by ASD (NII)/DOD CIO, the component will make final decision on details of the tailored plan subject to minimums specified in the TISP procedures linked from the CJCSI 6212 resource page and any special needs identified by the J-6 for the I&S certification process.

Representation

Representations for the DoDAF products may be drawn from many diagramming techniques including:

- tables,
- IDEF,
- Entity-Relationship Diagrams (ERDs),
- UML,
- SysML,

and other custom techniques depending on the product, tool used, and contractor/customer preferences. There is a UPDM (Unified Profile for DoDAF and MODAF) effort within the OMG to standardize the representation of DoDAF products when UML is used.

DoDAF generically describes in the representation of the artifacts to be generated, but allows considerable flexibility regarding the specific formats and modeling techniques. The DoDAF deskbook [12] provides examples in using traditional systems engineering and data engineering techniques, and secondly, UML format. DoDAF proclaims latitude in work product format, without professing one diagramming technique over another.

In addition to graphical representation, there is typically a requirement to provide metadata to the Defense Information Technology Portfolio Repository (DITPR) or other architectural repositories.

Meta-Model

DoDAF has a Meta-Model underpinning the framework, defining the types of modelling elements that can be used in each view and the relationships between them.

DoDAF versions 1.0 thru 1.5 used the CADM meta-model, which was defined in IDEF1X (then later in UML) with an XML Schema derived from the resulting relational database.

From version 2.0, DoDAF has adopted the IDEAS Group foundation ontology as the basis for its new meta-model. This new meta-model is called "DM2"; an acronym for "DoDAF Meta-Model".

Diagram of DoDAF V2.0 Meta Model DM2's Three Levels. [13]

DM2 metamodel 2.02 [14] (Enterprise Architect file)

Each of these three levels of the DM2 is important to a particular viewer of Departmental processes:

1. The conceptual level or Conceptual Data Model (CDM) defines the high-level data constructs from which Architectural Descriptions are created in non-technical terms, so that executives and managers at all levels can understand the data basis of Architectural Description. Represented in the DoDAF V2.0 DIV-1 Viewpoint.
2. The Logical Data Model (LDM) adds technical information, such as attributes to the CDM and, when necessary, clarifies relationships into an unambiguous usage definition. Represented in the DoDAF V2.0 DIV-2 Viewpoint.
3. The Physical Exchange Specification (PES) consists of the LDM with general data types specified and implementation attributes (e.g., source, date) added, and then generated as an XSD. Represented in the DoDAF V2.0 DIV-3 Viewpoint.[4]

The purposes of the DM2 are:

1. Establish and define the constrained vocabulary for description and discourse about DoDAF models (formerly “products”) and their usage in the 6 core processes
2. Specify the semantics and format for federated EA data exchange between:architecture development and analysis tools and architecture databases across the DoD Enterprise Architecture (EA) Community of Interest (COI) and with other authoritative data sources
3. Support discovery and understandability of EA data:
    1. Discovery of EA data using DM2 categories of information
    2. Understandability of EA data using DM2’s precise semantics augmented with linguistic traceability (aliases)

4. Provide a basis for semantic precision in architectural descriptions to support heterogeneous architectural description integration and analysis in support of core process decision making.[4]

The DM2 defines architectural data elements and enables the integration and federation of Architectural Descriptions. It establishes a basis for semantic (i.e., understanding) consistency within and across Architectural Descriptions. In this manner, the DM2 supports the exchange and reuse of architectural information among JCAs, Components, and Federal and Coalition partners, thus facilitating the understanding and implementation of interoperability of processes and systems. As the DM2 matures to meet the ongoing data requirements of process owners, decision makers, architects, and new technologies, it will to a resource that more completely supports the requirements for architectural data, published in a consistently understandable way, and will enable greater ease for discovering, sharing, and reusing architectural data across organizational boundaries.[4]

To facilitate the use of information at the data layer, the DoDAF describes a set of models for visualizing data through graphic, tabular, or textual means. These views relate to stakeholder requirements for producing an Architectural Description.[4]

Relationship to other architecture frameworks

The UPDM (Unified Profile for DoDAF and MODAF) is an OMG initiative to standardize UML and SysML usage for USA and UK defense architecture frameworks. In addition, the multi-national IDEAS Group, which is supported by Australia, Canada, Sweden, UK, USA, with NATO observers, has launched an initiative to develop a formal ontology for enterprise architectures.

Commercial Products Support

Several commercial products have support for DODAF, including the following.

- CMD2 by Casewise - the industry's first DoDAF 2.0 compliant and DM2 conformant solution. [15]
- Atego's Artisan Studio with UPDM, DoDAF and MODAF. [16]
- Elements Repository produced by Enterprise Elements, Inc.. [17]
- MagicDraw with UPDM plugin supporting DoDAF/MODAF. [18]
- MEGA International, MEGA Suite [19], with a fully integrated "MEGA for DoDAF" [20] supporting DoDAF 2.0, in addition to MEGA for NAF [21]
- System Architect (software)
- Sparx Enterprise Architect, Sparx Systems [22], with a plugin for UPDM 2.0 [23] (supports DoDAF 2.0 and MODAF 1.2).
- IServer produced by Orbus Software. Orbus Software [24]
- MDWorbench from Sodius - a solution for interoperabilitiy of DoDAF models to exchange models, views and diagrams between various authoring tools. Sodius web page [25]
- Open Text Corporation ProVision Modeling Tool with DODAF 2.0 built in. [Opentext Software [26]]

References

[1] DoD (2007) DoD Architecture Framework Version 1.5 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_I.pdf). 23 April 2007

[2] (reference: Zachman Framework)

[3] "Architecture Framework FAQ" (http://architectureframework.com/faq/). . Retrieved 2007-08-07.

[4] "DoDAF Meta Model (DM2)" (http://dodcio.defense.gov/dodaf20/dodaf20_dm2.aspx). .

[5] DoDAF 1.5 is presented in three volumes and a deskbook:

- DoDAF 1.5 Volume 1 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_I.pdf) - Provides definitions, guidelines, background material.
- DoDAF 1.5 Volume 2 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_II.pdf) - Describes each architecture product.
- DoDAF 1.5 Volume 3 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_III.pdf) - Provides the architecture data description.
- DoDAF 1.0 Deskbook (https://acc.dau.mil/CommunityBrowser.aspx?id=31667&lang=en-US) - Provides supplementary "how to" information relating to architectures. The DODAF architecture documents were updated on April 23, 2007 to version 1.5. Currently the Deskbook, which is from February 9, 2004, has not been updated. This link is only to the Final Draft version August 30, 2003 - not the Feb 9, '04 version

[6] DoD CIO Memo Releasing DoDAF 2.0 (http://jitc.fhu.disa.mil/jitc_dri/pdfs/dodafv2_28may09.pdf)

[7] DoD CIO DoDAF Website (http://dodcio.defense.gov/dodaf20.aspx)

[8] Diagram of DoDAF V2.0 Viewpoints

[9] Evolution of DoDAF V1.5 Views to DoDAF V2.0 Viewpoints

[10] Mapping of DoDAF V1.5 Views to DoDAF V2.0 Viewpoints

[11] DoDAF V1.5 Products Matrix

[12] https://acc.dau.mil/GetAttachment.aspx?id=31667&pname=file&lang=en-US&aid=28906

[13] http://dodcio.defense.gov/portals/0/Images/dodaf20/DM2s_3_levels.gif

[14] http://dodcio.defense.gov/dodaf20/dodaf20_logical.aspx

[15] http://www.casewise.com

[16] http://www.atego.com/products/artisan-studio-architect-enterprise-edition/

[17] http://www.enterprise-elements.com

[18] http://www.magicdraw.com/updm_plugin

[19] http://www.mega.com/en/c/product

[20] http://www.mega.com/en/c/product/p/modeling/p2/dodaf-implementation

[21] http://www.mega.com/en/c/product/p/modeling/p2/naf-implementation

[22] http://www.sparxsystems.com

[23] http://www.sparxsystems.com/products/mdg/tech/updm/index.html

[24] http://www.orbussoftware.com/

[25] http://www.sodius.com/

[26] http://www.opentext.com/2/global/products/products-business-process-management/products-opentext-provision-for-enterprise-architecture.htm/

Further reading

- Dennis E. Wisnosky and Joseph Vogel (2004). *Dodaf Wizdom: a Practical Guide to Planning, Managing and Executing Projects to Build Enterprise Architectures using the Department of Defense Architecture Framework*. Wizdom Systems, Inc., 2004. ISBN 1-893990-09-5.
- DoDAF V2.02 (http://dodcio.defense.gov/dodaf20.aspx) - This is the official and current version for the Department of Defense Architecture Framework.
- DoDAF V2.0 was approved May 28, 2009. It is presented in three volumes and a Journal:
  - DoDAF V2.0 Volume 1 : Introduction, Overview, and Concepts - Manager's Guide (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF V2 - Volume 1.pdf)
  - DoDAF V2.0 Volume 2 : Architectural Data and Models - Architect's Guide (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF V2 - Volume 2.pdf)
  - DoDAF V2.0 Volume 3 : DoDAF Meta-model Physical Exchange Specification - Developer's Guide (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF V2 - Volume 3.pdf)

- DoDAF V2.0 Journal : The electronic interface for DoDAF support (http://dodcio.defense.gov/dodaf20/dodaf20_journal.aspx).

- Architectural Resources cited in DoDAF V2.0. A number of architecture resources exist which serve as sources for guidelines that should be consulted while building architectural views. Some architecture resources require Secret Internet Protocol Router Network (SIPRNET) access.

DoDAF V2.0 Resources (http://dodcio.defense.gov/dodaf20.aspx)

- DoD IEA: Department of Defense Information Enterprise Architecture (DoD IEA) (http://dodcio.defense.gov/Home/Initiatives/DIEA.aspx) Defines the key principles, rules, constraints and best practices to which applicable DoD programs, regardless of Component or portfolio, must adhere in order to enable agile, collaborative net-centric operations. The DoD IEA provides the guidelines and rules that the architect must keep in mind in the architecture development effort.
- DARS: DoD Architecture Registry System (DARS) (https://dars1.army.mil) DARS is the DoD registry of segment and solution architectures comprising the federated DoD enterprise architecture. To discover architectures that exist, or may be in development. Depending on the purpose and scope, an architect may search and discover Architectures that overlap the scope and purpose of the architecture effort. To register metadata about architectures that are being developed, or currently exist.
- DITPR: DoD Information Technology Portfolio Repository (DITPR) (https://www.dadms.navy.mil/) The official unclassified DoD data source for Federal Information Security Management Act (FISMA), EAuthentication, Portfolio

Management, Privacy Impact Assessments, the inventory of MC/ME/MS systems, and the registry for systems under DoDI 5000.2. The Systems metadata from the Architecture can be used to populate DITPR with new or updated information. DITPR can also populate the architecture's Systems metadata, particularly on systems that interface with systems described in the architecture, but are not part of the scope of the architecture.

- DISR: DoD Information Technology Standards and Profile Registry (DISR) (https://disronline.disa.mil) Online repository for a minimal set of primarily commercial IT standards. The DISR can be used to populate the Standards models (StdV-1 and StdV-2) of the Architecture. Conversely, the Standards Models can identify additional or new standards that need to be added to DISR.
- JCPAT: (SIPRNet) Joint C4I Program Assessment Tool (JCPAT) (http://jcpat.ncr.disa.smil.mil/JECOweb.nsf) Formally assess systems and capabilities documents (Initial Capabilities Document, Capability Development Document, and Capability Production Document) for Joint Staff interoperability requirements certification and is the ITS/NSS Lifecycle Repository and the archives. The ICD, CDD, and CPD contain architecture information. As the architecture development progresses, the collected architecture information can be extracted and reported in the ICD, CDD, and the CPD. In addition, the architecture information can be within with the Enhanced-Information Support Plan (E-ISP) tool, a part of the JCPAT toolset.
- JCSFL: Joint Common System Function List (JCSFL) (https://us.army.mil/suite/page/419489) A common lexicon of

systems/service functionality supporting joint capability. The JCSFL is provided for mapping functions to supported activities and the systems or services that host them. Chairman of the Joint Chiefs of Staff Instruction (CJCSI) 6212.01E prescribes the JCSFL for use in developing a common vocabulary for architecture development. Use the taxonomy to align or extend system functions within the architecture being developed

- KM/DS: (SIPRNet) Knowledge Management/ Decision Support (KM/DS) (https://jrockmds1.js.smil.mil/guestjrcz/gbase.guesthom) The KM/DS tool will be used by DoD components to submit documents and comments for O-6 and flag reviews, search for historical information, and track the status of documents. Supporting the JCIDS approval process, the

documents that are necessary for Milestone Decisions have architecture information. As the architecture development progresses, the collected architecture information can be extracted and reported in the required

documents.

- DoD MDR: Metadata Registry (http://metadata.dod.mil) The DoD Metadata Registry and Clearinghouse provides software developers access to data technologies to support DoD mission applications. Through the Metadata Registry and Clearinghouse, software developers can access registered XML data and metadata components, database segments, and

reference data tables and related metadata information The Resource Flows and Physical Schemas from the Architecture can be used to populate the Metadata Registry.

- NAERG: Naval Architecture Elements Reference Guide (NAERG) (https://stalwart.spawar.navy.mil/naerg/) A standard terms of reference for the Navy and Marine Corps. The Architecture Elements represent the critical taxonomies requiring

concurrence and standardization for an integrated architecture. They comprise the lexicon for the three views of the architecture framework, the operational (OV), system (SV) and technical standards (TV) views. The use of the critical taxonomies is a step to ensuring integration of systems within a system of systems and alignment of information technology (IT) functionality to mission and operational needs. The data contained in each element of the Architecture list shall be used for overall architecture framework development, programmatic research, development, and acquisition activities, and related integration and interoperability and capability assessments. It will be updated through review periods to support DoN Program Objective Memorandum (POM) efforts and to reflect changes mandated by DoD, technology improvements, and other factors.

- Service Registry: Service Registry. Select the “NCES Service Discovery” button (http://metadata.dod.mil) The Service Registry provides enterprise-wide insight, control and leverage of an organization's services. It captures service descriptions and makes them discoverable from a centrally managed, reliable, and searchable location. The Services

metadata from the Architecture effort can be used to populate the Service Registry in the process of developing the solution.

- UJTL: Universal Joint Task List (UJTL) (http://www.dtic.mil/doctrine/jel/cjcsd/cjcsm/m350004c.pdf) The Universal Joint Task List from the Chairman of the Joint Chiefs of Staff Manual 3500.04C (CJCSM) serves as a common language and

common reference system for joint force commanders, combat support agencies, operational planners, combat developers, and trainers to communicate mission requirements. It is the basic language for development of a joint mission essential task list (JMETL) or agency mission essential task list (AMETL) that identifies required capabilities for mission success. Use the taxonomy to align or extend operational activities within the architecture being developed.

External links

- DoDAF V2.02 (http://dodcio.defense.gov/dodaf20.aspx) - This is the official and current version for the Department of Defense Architecture Framework.
- Printable version of DoDAF V2.0 Volume 1 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_I.pdf)
- Printable version of DoDAF V2.0 Volume 2 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_II.pdf)
- Printable version of DoDAF V2.0 Volume 3 (http://dodcio.defense.gov/Portals/0/Documents/DODAF/DoDAF_Volume_III.pdf)
- DoDAF Promulgation Memo May 28, 2009 (http://jitc.fhu.disa.mil/jitc_dri/pdfs/dodafv2_28may09.pdf) - The DODAF Policy Directive which mandates that all DoD architectures developed after 05/28/2009 (or the next major system release) must be DODAF V2.0 compliant.

- DoDAF Promulgation Memo Feb 9, 2004 (http://www.aitcnet.org/dodfw/DoDAF_Memo.pdf) - The DODAF Policy Directive which mandates that all DoD architectures approved after 12/01/03 must be DODAF V1.5 compliant. (OBSOLETED)
- DoDAF section of Architecture Framework Forum (http://www.architectureframework.com/frameworks/dodaf/) Information resource dedicated to DoDAF as it relates to other architecture frameworks (e.g., MODAF, TOGAF, Zachman).
- DoD BEA v7.0 (http://www.bta.mil/products/BEA_7_0/BEA/html_files/home.html) - DoD Business Enterprise Architecture BEA 7.0 (March 2010)
- Two Presentations (http://www.integrated-ea.com/Previous-Years) on DoDAF 2.0 from Integrated EA Conferences 2008 and 2009

MODAF

United Kingdom Ministry of Defence	
Department overview	
Formed	1964 (As modern department)
Jurisdiction	United Kingdom
Headquarters	Whitehall, Westminster, London[1]
Employees	over 80,000 civilian staff[2]
Annual budget	£35.165 billion (2009/10)[3]
Minister responsible	The Rt Hon Philip Hammond, MP, Secretary of State for Defence
Department executives	General Sir David Richards, Chief of the Defence Staff Ursula Brennan, Permanent Secretary
Website	
http://www.mod.uk	

The **British Ministry of Defence Architecture Framework** (**MODAF**) is an Architecture Framework which defines a standardised way of conducting Enterprise Architecture, originally developed by the UK Ministry of Defence.

Initially the purpose of MODAF was to provide rigour and structure to support the definition and integration of MOD equipment capability, particularly in support of Network Enabled Capability (NEC).

More recently, MOD has additionally been using MODAF to underpin the use of the Enterprise Architecture approach to the capture of the information about the business to identify the processes and resources required to deliver the vision expressed in the strategy.

Overview

MODAF is an internationally recognised enterprise architecture framework developed by the MOD to support Defence planning and change management activities. It does this by enabling the capture and presentation of information in a rigorous, coherent and comprehensive way that aids the understanding of complex issues, thereby providing managers with the key factors they should consider when making decisions about changes to the business. It is used extensively in Defence acquisition to support systems engineering, particularly in support of Network Enabled Capability (NEC], "which is about the coherent integration of sensors, decision-makers, weapon systems and support capabilities to achieve the desired effect."

With the publication of the MOD Information Strategy (MODIS)[4] and its Enterprise Architecture (EA) Sub-Strategy, the MOD has recognised the utility of EA to support business improvement. MODAF is central to the use of EA in MOD.

MODAF is managed and maintained by staff working for the MOD's Chief Information Officer[5](CIO), as part of their role to provide Information Policy and Standards. Additional support is provided by the MOD's System Engineering and Integration Group, as part of their role in developing the System of Systems Approach (SOSA),[6] a common set of principles, rules, and standards to enable the delivery of better interoperability between systems.

MOD is working closely with its International Allies to ensure coherence with their architecture frameworks to enable the sharing of information about capabilities fielded in coalition operations in-order to support

interoperability. MODAF was developed from the US Department of Defense Architecture Framework (DoDAF) version 1.0, but has been extended and modified to meet MOD requirements by the addition of Strategic, Acquisition and Service Oriented Viewpoints and the provision of the M3.

MODAF version 1.0 was released in 2005, following development work by the MODAF Partners, a collaborative team of MOD staff and contractors from a number of industry partners and has been continuously improved since Version 1.0, the latest release, version 1.2.004, was released in May 2010

History

MODAF was initially developed for MOD from two parallel work strands, an MOD-funded research programme undertaken by QinetiQ (formerly the Defence Evaluation Research Agency) and a separate DoDAF-based development by MODAF Partners, a consortium of Cornwell Management Consulting (now Serco Group) and PA Consulting Group with Model Futures [7] providing the technical input, and extended by other key suppliers such as Logica and Vega through work for the MOD Integration Authority (as of April 2008 the System Engineering Integration Group (SEIG)). The initial version of MODAF combined the metamodel developed from the QinetiQ research programme and the views developed by MODAF Partners.

Framework

MODAF is a set of templates (called “Views”) that provide a standard notation for the capture of information about a business in order to identify ways to improve the business. Each MODAF View offers a different perspective on the business to support different stakeholder interests, presented in a format, usually graphical, that aids understanding of how a business operates.

The Views are divided into seven categories (Viewpoints):

- Strategic Viewpoint (StV) defines the desired business outcome, and what capabilities are required to achieve it;
- Operational Viewpoint (OV) defines (in abstract rather than physical terms) the processes, information and entities needed to fulfil the capability requirements;
- Service Orientated Viewpoint (SOV) describes the services, (i.e. units of work supplied by providers to consumers), required to support the processes described in the operational Views;
- Systems Viewpoint (SV) describes the physical implementation of the Operational and Service Orientated Views and, thereby, define the solution;
- Acquisition Viewpoint (AcV) describes the dependencies and timelines of the projects that will of deliver the solution;
- Technical Viewpoint (TV) defines the standards that are to be applied to the solution;
- All Viewpoint (AV) provides a description and glossary of the contents of the architecture.

The relationship between the data in the MODAF Views is defined in the MODAF Meta Model, known as the M3. The M3 provides a logical structure for the storage of the data in a database and, subsequent, provides the necessary coherence for the data to be shared with other MODAF architectures.

Functionality of Framework

In MOD, MODAF has primarily been used in acquisition domains, programmes and delivery teams to support the delivery of military capability, particularly NEC. A number of MODAF architectures directly support operations in Afghanistan. In addition, MODAF is widely used by its industry partners, such as BAE Systems, Thales, Lockheed Martin, Boeing and Serco. It is also used by other government departments and agencies, such as GCHQ, and external bodies such as the National Air Traffic Services (NATS). MODAF is used by the Swedish Armed Forces to support the development of military capability, and it has been adapted by NATO to form the core of the NATO

Architecture Framework (NAF).

Harmonisation

MODAF will continue in its current form for the foreseeable future. However, MOD is working closely with the United States Department of Defense, the Canadian Department of National Defence, the Australian Department of Defence, and the Swedish Armed Forces to develop the International Defence Enterprise Architecture Specification (IDEAS). Although the focus for IDEAS has been the ability to provide a mechanism to better enable the exchange of architecture information between Nations, the IDEAS Management Group are also actively considering how their architecture frameworks should converge, perhaps into a single unified architecture framework.

Tools and Tooling

MOD is “agnostic” about which software tools should or should not be used to develop MODAF architectures. The key requirement is that they should correctly implement the M3[8] with downloads in Sparx Systems Enterprise Architect; HTML and XMI formats. to provide a coherent structure against which architecture information can be exchanged. A number of tools offer this functionality.

MOD has been working with the Object Management Group (OMG) to develop the Unified Profile for DoDAF and MODAF (UPDM), a standard specification that provides an accurate representation of the M3 for tools based on the Unified Modelling Language (UML) or Systems Modelling Language (SysML)

Terminology

An "architectural framework" or "architecture framework" is a specification of how to organise and present architectural models. An architectural framework consists of a standard set of views, which each have a specific purpose.

An "architectural description" is a contiguous, coherent model of an enterprise. An architectural description comprises "architectural products". MODAF is not an architectural description.

A "view" is a specification of a way to present an aspect of the enterprise. Views are defined with one or more purposes in mind - e.g., showing the logical topology of the enterprise, describing a process model, defining a data model, etc.

An "architectural product" is a model of a particular aspect of the enterprise. An architectural product conforms to a "view".

A "viewpoint" is a collection of "views." Viewpoints are usually categorized by domain - e.g., in MODAF there are seven viewpoints.

Applications

Although originally developed by the UK Ministry of Defence, MODAF is the standard architecture framework for other organisations, such as:

- GCHQ
- Swedish Armed Forces
- National Air Traffic Services
- MooD International use MODAF Solutions (built using their MooD Technology) for the Defence and National Security Sectors - see http://www.moodinternational.com/
- BAE Systems use MODAF on a number of internal programmes, most notably their TRAiDE environment
- EADS use MODAF as part of the Modelling and Simulation process for NetCOS their Synthetic Environment
- Thales Group use MODAF in their work for UK MOD

- Avolution use MODAF widely within the UK Defence and Security sectors
- Niteworks use MODAF extensively on problem solving and warfighter experimentation for the UK MOD - See http://www.niteworks.net
- Centric Labs use MODAF as part of the Modelling process for the Swedish Armed Forces
- BAA Limited - see http://www.centriclabs.se/
- Model Futures Ltd. uses MODAF, and were part of the original team that developed the framework for MOD - see http://www.modelfutures.com

In addition, revision 3 of the NATO Architecture Framework (NAF) is identical to MODAF at its core, but extends the framework by adding views for Bandwidth Analysis, SOA and Standards configurations.

MODAF is also the basis for other frameworks such as TRAK, a domain-free framework, which is based on MODAF 1.2

References

[1] 51°30′14″N 0°7′30″W
[2] "Defence in the Community" (http://www.mod.uk/DefenceInternet/AboutDefence/Organisation/KeyFactsAboutDefence/DefenceInTheCommunity.htm). Ministry of Defence. . Retrieved 22 March 2010.
[3] "Defence Spending" (http://www.mod.uk/DefenceInternet/AboutDefence/Organisation/KeyFactsAboutDefence/DefenceSpending.htm). Ministry of Defence. . Retrieved 22 March 2010.
[4] Error pages (http://www.mod.uk/DefenceInternet/AboutDefence/CorporatePublications/PolicyStrategyandPlanning/ModInformationStrategy2009.htm)
[5] House of Commons Hansard Written Answers for 13 Jan 2009 (pt 0004) (http://www.publications.parliament.uk/pa/cm200809/cmhansrd/cm090113/text/90113w0004.htm)
[6] (http://www.aof.mod.uk/aofcontent/tactical/engineering/content/systems_guidance/se_systemofsystems.htm)
[7] http://www.modelfutures.com/
[8] Ministry of Defence | About Defence | Corporate Publications | Information Management | MODAF | MODAF Meta Model (http://www.mod.uk/DefenceInternet/AboutDefence/CorporatePublications/InformationManagement/MODAF/ModafMetaModel.htm)

External links

- MOD Information Strategy (MODIS) 2009 (http://www.mod.uk/DefenceInternet/AboutDefence/CorporatePublications/PolicyStrategyandPlanning/ModInformationStrategy2009.htm)
- DE&S Systems of Systems Approach (SOSA) (http://www.aof.mod.uk/aofcontent/tactical/engineering/content/systems_guidance/se_systemofsystems.htm)
- MOD Site for MODAF (http://www.mod.uk/DefenceInternet/AboutDefence/WhatWeDo/InformationManagement/MODAF)
- MODAF Meta-Model (http://www.mod.uk/DefenceInternet/AboutDefence/CorporatePublications/InformationManagement/MODAF/ModafMetaModel.htm)
- Zachman International (http://www.zachmaninternational.us/)
- Integrated EA (http://www.integrated-ea.com/Previous-Years) Various Presentations on MODAF from Integrated EA Conferences 2008-2010

NATO Architecture Framework

The **NATO Architecture Framework** is an Enterprise Architecture framework by the NATO derived from the DoDAF Enterprise architecture.

The current NATO C3 System Architecture Framework v2 (NAF v2), issued by NATO in September 2004 provides guidance on describing communication and information systems. Revision 3 of the NATO Architecture Framework (NAF), promulgated in November 2007, is identical to MODAF at its core, but extends the framework by adding views for Bandwidth Analysis, SOA and standard configurations.

The seven views are:

- NATO All View (NAV)
- NATO Capability View (NCV)
- NATO Operational View (NOV)
- NATO Service-Oriented View (NSOV)
- NATO Systems View (NSV)
- NATO Technical View (NTV)
- NATO Programme View (NPV)

Each view has a set of subviews.

NAF Meta-Model (NMM)

NAF uses the same meta-model as MODAF. The meta-model is documented in chapter 5 of the NATO Architecture Framework. Version 3.1 of the NMM is identical to v1.2.003 of the MODAF Meta-Model

Using the UK Ministry of Defence's MODAF Documentation for NAF Work

The documentation of the NAF Rev 3 views (Chapter 4) does not always align well with the NAF Meta Model (Chapter 5). This is particularly the case with some of the examples, which are based on DoDAF version 1.0. Some NAF users find it useful to first of all refer the official MODAF Documentation - [1]. This is a useful strategy, as the MOD documentation can be somewhat easier to follow, and NAF and MODAF share a common meta-model. However, there are some issues to consider:

- The MODAF documentation covers version 1.2.004 of the meta-model which has some small differences from v3.1 of the NAF Meta-Model (which was a NATO release of v1.2.001 of the MODAF Meta-Model). At the time of writing, plans were in place to update the NAF Meta-Model to correspond with MODAF v1.2.004. At any time though, there is a chance of NAF being behind MODAF as updates are made.
- The SOA views differ significantly between MODAF and NAF. In particular, MODAF handles in service orchestration in the Operational Viewpoint (OV-5) whereas NAF uses NSOV-5
- MODAF does not have a TV-3 view

In most other respects, the MODAF documentation is fit for use in developing NAF architectures.

External links

- NATO C3 Board [2]
- NATO Architecture Framework v3 [3]
- The initial work done on the bandwidth and frequency view in NAF v3 has since been further investigated by UK MOD CIO-J6 as part of the Spectrum Management work for CCEB [4]
- NATO Presentation on NAF [5] by Frits Broekema from Integrated EA Conference 2009
- Summary of changes from NAF v2 to v3 [6]

References

[1] http://www.mod.uk/modaf
[2] http://www.nhqc3s.nato.int/
[3] http://www.nhqc3s.nato.int/ARCHITECTURE/_docs/NAF_v3/ANNEX1.pdf
[4] http://www.modaf.com/file_download/41/Spectrum+Architecture+v1_1.pdf
[5] http://www.integrated-ea.com/Previous-Years
[6] http://trak-community.org/index.php/wiki/NAF:NAF_3.0

AGATE Architecture Framework

AGATE (**A**telier de **G**estion de l'**A**rchi**TE**cture des systèmes d'information et de communication) is a framework for modeling computer or communication systems architecture.

It is promoted by the Délégation Générale pour l'Armement (DGA), the French government agency which conducts development and evaluation programs for weapon systems for the French military. All major DGA weapons and information technology system procurements are required to document their proposed system architecture using the set of views prescribed in AGATE.

AGATE is similar to DoDAF, promoted by U.S. Department of Defense (DoD) or MODAF, promoted by UK Ministry of Defence (MoD). However, the fact that the AGATE reference is only available in French seems to limit its use to French weapons systems procurement.

Scope

AGATE defines architectural views for systems and systems of systems, covering:

- Stakes and objectives of the system
- Description of the related organizations
- processes and information flows
- Security requirements, in compliance with DGA policy
- Services of the system, and traceability with operational needs
- Logical architecture of the system
- Physical architecture of the systems, and hardware and software products used in this architecture
- Life cycle of the system

AGATE Views

An AGATE model is organized into 5 views:

- Stakes, Objectives, and context about the system
- Business architecture: describes organizations and Business processes managed by the modelized Information system
- Service-oriented architecture: describes the Services of the system.
- Logical architecture of the system
- Physical architecture of the systems, and hardware and software products used in this architecture

Representation

The AGATE meta-model is defined using a UML representation.

Visio elements for the AGATE representation are provided by the DGA.

The French government is planning to port AGATE representation to Telelogic System Architect[1]

History

The first DGA initiative for a standardized French architecture framework was initialized in July 2001, under the acronym **AMAC**. The denomination was changed to **AGATE** in November of the same year.[2]

- March 2004: Version 2.1
- December 2005: Version 3

References

[1] "Plan d'actions de l'initiative 2007" (http://synergies.modernisation.gouv.fr/IMG/pdf/IT06-PlanActions-20070401-V1.0.pdf). synergies.modernisation.gouv.fr. 2007-06-01. . Retrieved 2007-07-27.

[2] *Manuel de référence AGATE V3*, December 2005, *Eléments de départ AGATE et Origine*

External links

- AGATE page on French Defense procurement website (http://www.achats.defense.gouv.fr/article33349)
- short presentation of AGATE on French Defense procurement portal (http://www.ixarm.com/IMG/pdf/3-volets_Agate-2.pdf)

Government frameworks

Government Enterprise Architecture

The **Government Enterprise Architecture** (also known as the **GEA**) is an initiative of the Queensland Chief Information Office (QGCIO) in Australia and provides a guiding framework for Queensland Government's development, use, and management of information technology (IT) assets. The framework is designed to support the development of better services for Queenslanders, more efficient and effective use of ICT in government, and effective partnering with the private sector.

Compliance with the GEA is a requirement of the Queensland Government's Financial Management Standard.

History

The former Department of Communication and Information, Local Government, Planning and Sport commenced a project in early 1999 to define what was called at the time a Government Information Architecture with the aim of improving the interoperability of communication and information systems and the sharing of information resources across Queensland Government agencies.

Loosely based on the META Group Enterprise Architecture Service methodology the first version was published in May 2001.

After an internal review of IT management across the Queensland Government in 2005 this original effort was replaced by the expanded GEA. At the same time the notion of enterprise architecture became enshrined in legislation in the form of the Financial Management Standard. This became one of the first known examples of enterprise architecture being explicitly mandated by a government body.

External links

- GEA [1]
- Queensland Chief Information Office [2]

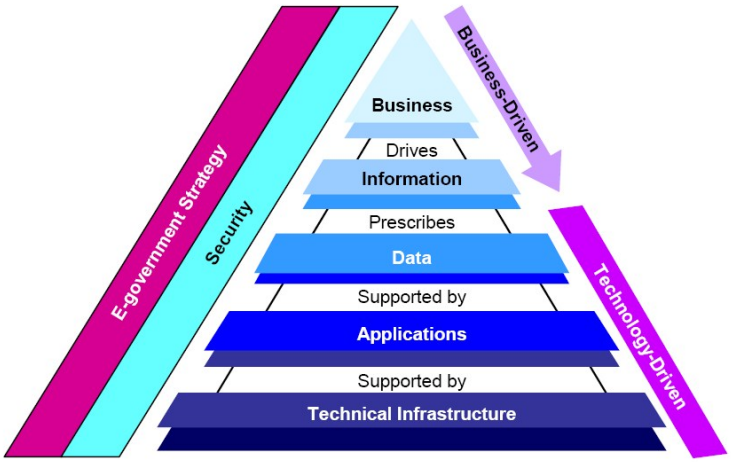
Government Enterprise Architecture is a guiding framework which presents guidelines and architecture principles from business services and technology services perspective defining the principles of interoperability between departments and better and efficient service delivery to the citizens and businesses in a country or state. Thus for GEA the current state assessment of each department is done to understand the business functions of each from G2G, G2C and G2B perspective and subsequently studying best practices from various countries the business process re-engineering if any is done. The business process re-engineering would involve adopting the best practices recommendations by the concerned individual departments and then implenting the technological parameters to deliver the current and new business practices through a common platform called the GEA to the citizens and services.

References

[1] http://www.qgcio.qld.gov.au/qgcio/architectureandstandards/qgea2.0/Pages/index.aspx
[2] http://www.qgcio.qld.gov.au

FDIC Enterprise Architecture Framework

FDIC Enterprise Architecture Framework is the Enterprise Architecture framework of the United States Federal Deposit Insurance Corporation (FDIC). A lot of the current article is about the Enterprise Architecture Framework developed around 2005, and currently anno 2011 out-of-date.



FDIC's Enterprise Architecture Framework from 2005.[1]

Overview

The FDIC's framework for implementing its Enterprise Architecture is based on Federal and industry best practices, including the Chief Information Officer (CIO) Council's Federal Enterprise Architecture Framework (FEAF) and the Zachman Framework for Enterprise Architecture. FDIC's framework has been tailored to emphasize security. The FDIC EA framework complies with the FEAF and highlights the importance of security to all other components of the architecture.[2]

The FDIC EA framework includes five components. The first component, the Business Architecture, focuses on FDIC's business needs. The next three components, the Data Architecture, Applications Architecture, and Technical Infrastructure Architectures, focus on the technological capabilities that support the business and information needs. The final component, the Security Architecture, focuses on specific aspects of interest to the Corporation that span the enterprise and must be integral parts of all other architectures.[2]

History

Historically, Federal agencies have managed IT investments autonomously. Until the new millennium, there has been little incentive for agencies to partner to effectively reuse IT investments, share IT knowledge, and explore joint solutions. A collective, government-wide effort, supported by the Federal CIO Council, utilizing the Federal Enterprise Architecture (FEA), has been undertaken in an effort to yield significant improvements in the management and reuse of IT investments, while improving services to citizens, and facilitating business relationships internally and externally. The FEA is a business-based framework that provides the Office of Management and Budget (OMB) and Federal agencies a way to monitor, analyze, and control Federal IT investments.[3] The FDIC first realized the value of EA in 1997, when two business executives had to reconcile data that had come from different systems for a high-profile report to the banking industry. The FDIC's first EA blueprint was published in December 2002.[4]

In 2004 the FDIC received a 2004 Enterprise Architecture Excellence Award from the Zachman Institute for Framework Advancement (ZIFA) for its initiative to manage corporate data collaboratively. John Zachman, an expert on enterprise architecture, founded ZIFA, a network of information professionals supporting enterprise

architecture's role in helping organizations operate from a corporate perspective.[5]

EA framework topics

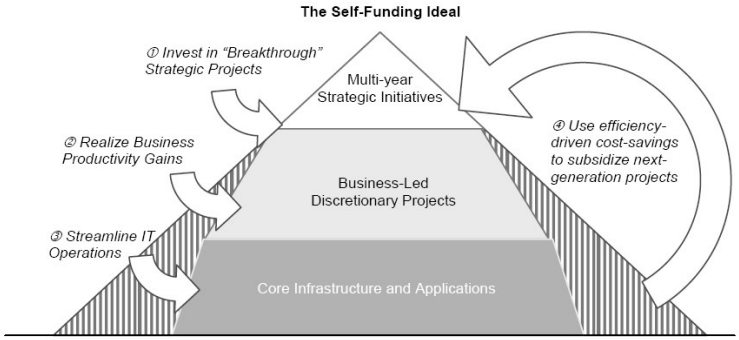
FDIC EA framework

The FDIC EA framework from 2005 included five components.

- *Business Architecture* : The Business Architecture describes the activities and processes performed by the Corporation to achieve its mission and to realize its vision and goals. Developing the Business Architecture is the first step in creating an Enterprise Architecture (EA) that links the Corporation's business needs to its Information Technology (IT) environment. Maximizing IT support for these requirements will optimize Corporate performance.[2]
- *Data Architecture* : The Data Architecture describes the activities required to obtain and maintain data that supports the information needed by the Corporation's major business areas. Data and information are different. Data is the foundation of information. Data is the raw material that is processed and refined to generate information. Information consists of a collection of related data that has been processed into a form that is meaningful to the recipient.[2]
- *Applications Architecture* : The Applications Architecture describes the major types of applications that manage data to produce the information needed to support the activities of the Corporation. The Applications Architecture provides a framework that enables the migration from the current applications catalog and software development environment to the target integrated applications, development and engineering environments. The target architecture promotes the use of commercial and government off-the-shelf products, consolidating applications, where applicable, and the use of emerging technologies where appropriate.[2]
- *Technical Infrastructure Architecture* : The IT infrastructure provides access to application systems and office automation tools used in performance of the business processes. The Corporation places high priority on maintaining a consistent, available, and reliable technical infrastructure. The Technical Architecture describes the underlying technology for the Corporation's business, data, and application processing. It includes the technologies used for communications, data storage, application processing, and computing platforms.[2]
- *Security Architecture* : The Security Architecture establishes a framework for integrating safeguards into all layers of the FDIC's Enterprise Architecture. The security architecture uses a risk management and information assurance strategy that provides access control, confidentiality, integrity, and non-repudiation for the Corporation's information and systems.[2]

Future IT development

The banking business model has become more complex, giving rise to financial instruments such as collateralized debt obligations (CDOs) and structured investment vehicles (SIVs) to manage risk. These instruments have created greater dependencies between the domestic and international financial markets. Financial institutions must, therefore,



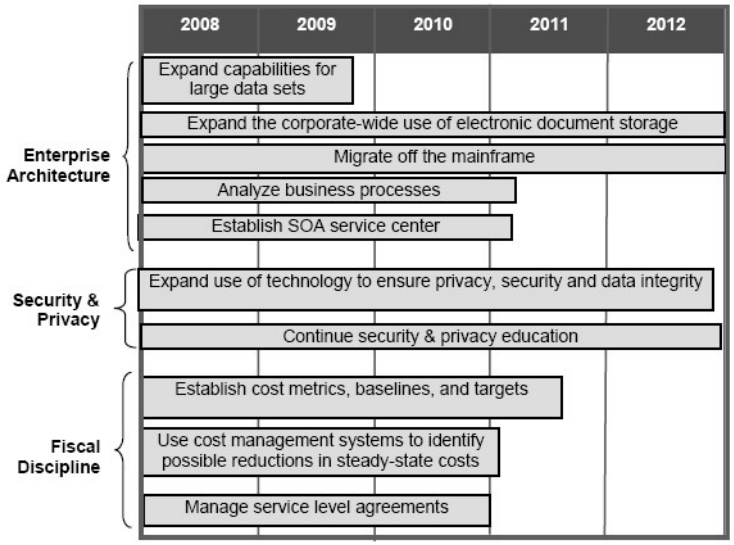
Self-Funding Model for Future IT Development, 2008.[6]

strike a balance between regulatory, legislative and banker concerns while appropriately managing risk.[6]

As cost savings are realized from a simplified IT environment and more efficient processes, the savings will be reinvested for IT improvements or accrue to the Corporation. This self-funding model is shown on the right. [6]

Five-year technology roadmap

The technology roadmap outlines the major initiatives for standardizing the IT environment and increasing IT's efficiency and effectiveness over the next five years. The initiatives were determined by various sources including business-side IT roadmaps, executive management planning meetings, client planning sessions, and client year-end reviews. The three major initiatives identified are enterprise architecture, security and privacy programs, and fiscal discipline.[6]



Five-Year Technology Roadmap, 2008.

The enterprise architecture initiative will focus on simplifying the environment to ensure stable and economical performance for mission-critical applications. Simplifying the environment to decrease costs will include activities, such as decreasing the number of application systems and migrating applications off the mainframe. Efficiencies will also be gained by expanding capabilities for manipulating large data sets and storing traditional paper-based files electronically. The SOA service center will manage code (or services) for all development teams to discover and use, which will save time and costs in application development, testing and deployment.[6]

The organization will continue to enhance IT security and privacy programs to address new and evolving risks by improving controls over sensitive data. In some cases, technology, such as scanning outgoing e-mail for sensitive information and encrypting removable storage devices, can mitigate potential risks. The other cornerstone of mitigating risk is educating employees of emerging security and privacy issues.[6]

Lastly, in order to continue sound fiscal discipline and responsibility, the organization will establish IT baselines and metrics, study steady-state costs, manage service level agreements, and more judiciously choose new development projects. These three areas – enterprise architecture, security and privacy programs, and fiscal discipline – are shown below with the estimated time frames.[6]

References

[1] OIG (2005). Implementation of E-Government Principles (http://www.fdicoig.gov/reports05/05-018-508.shtml). May 2005
[2] Implementation of E-Government Principles (http://www.fdicoig.gov/reports05/05-018-508-figure1.shtml) AUDIT REPORT, Report No. 05-018, May 2005
[3] FDIC (2003). *Information Technology Strategic Plan 2004–2007* (http://www.iriboffice.ir/LinkClick.aspx?fileticket=up56ppcNQ4M=&tabid=246&mid=634)
[4] Gregg Kreizman, Cathleen E. Blanton (2005) " The FDIC Is Aligning IT to Business Through Enterprise Architecture (http://www.aea-dc.org/resources/2006-7-12-Gail-Verley-FDIC-EA-Business-Alignment-Gartner.pdf)" Gartner, Inc.
[5] FDIC Receives Technology Award (http://www.fdic.gov/news/news/press/2004/pr13104.html)
[6] CIO Council (2008) Information Technology Strategic Plan 2008–2013 (http://www.fdic.gov/about/strategic/it_plan/IT_Strategic_Plan_2.pdf), January 23, 2008.

Further reading

- Pallab Saha (2007). *Handbook of Enterprise Systems Architecture in Practice*. Chapter IX gives a detailed case study of the FDIC.

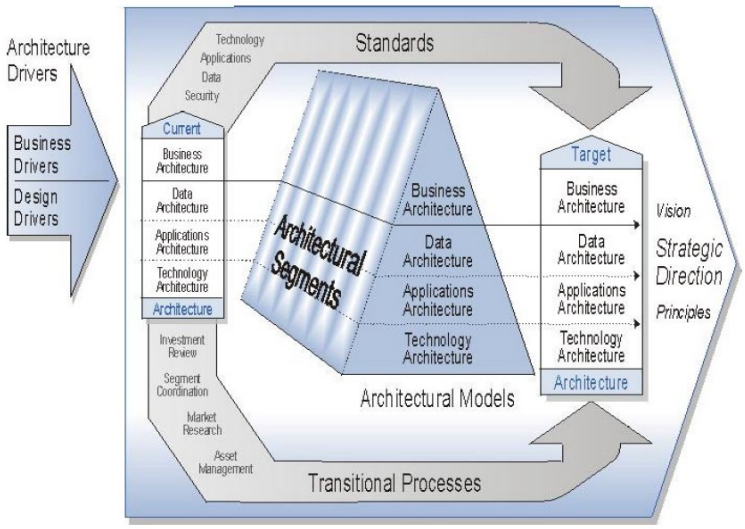
External links

- FDIC (http://www.fdic.gov/index.html) Homepage.

Federal Enterprise Architecture

A **federal enterprise architecture** (FEA) is the enterprise architecture of a federal government. It provides a common methodology for information technology (IT) acquisition, use, and disposal in the Federal government.[1]

Enterprise architecture (EA) is a management practice for aligning resources to improve business performance and help government agencies better execute their core missions. An EA describes the current and future state of the agency, and lays out a plan for transitioning from the current state to the desired future state. A federal enterprise architecture is a work in progress to achieve these goals.[3]



Structure of the U.S. "Federal Enterprise Architecture Framework" (FEAF) Components, presented in 2001.[2]

The U.S. federal enterprise architecture (FEA) is an initiative of the U.S. Office of Management and Budget that aims to comply with the Clinger-Cohen Act and provide a common methodology for IT acquisition in the United States federal government. It is designed to ease sharing of information and resources across federal agencies, reduce costs, and improve citizen services.[4]

History

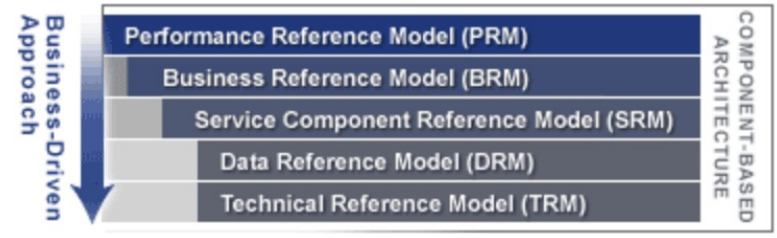
In September 1999, the Federal CIO Council published the "Federal Enterprise Architecture Framework" (FEAF) Version 1.1 for developing an Enterprise Architecture (EA) within any Federal Agency for a system that transcends multiple inter-agency boundaries. It builds on common business practices and designs that cross organizational boundaries, among others the NIST Enterprise Architecture Model. The FEAF provides an enduring standard for developing and documenting architecture descriptions of high-priority areas. It provides guidance in describing architectures for multi-organizational functional segments of the Federal Government.[2]

These federal architectural segments collectively constitute the federal enterprise architecture. In 2001, the Federal Architecture Working Group (FAWG) was sponsoring the development of Enterprise Architecture products for trade and grant Federal architecture segments. Method□a prescribed way of approaching a particular problem. As shown in the figure, the FEAF partitions a given architecture into business, data, applications, and technology architectures. The FEAF overall framework created that time, see image, includes the first three columns of the Zachman Framework and the Spewak's Enterprise Architecture Planning methodology.[2]

Reference models

The FEA is built using an assortment of reference models, that develop a common taxonomy and ontology for describing IT resources. These include the, (see image):

- performance reference model,
- business reference model,
- service component reference model,
- data reference model and
- technical reference model.



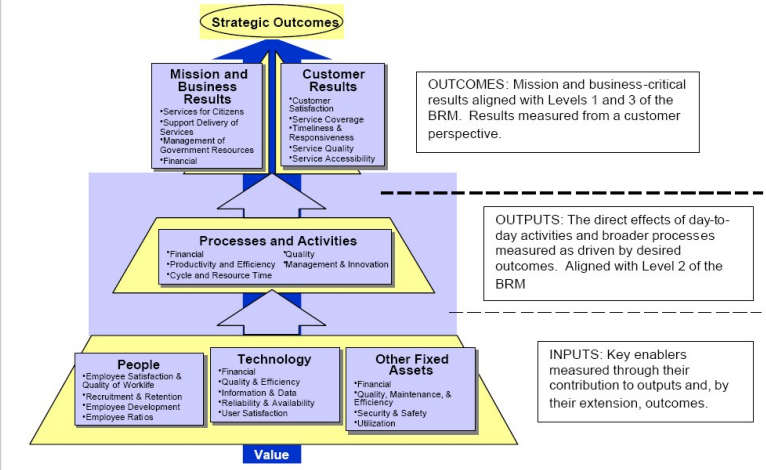
Federal Enterprise Architecture.[1]

It is designed to ease sharing of information and resources across federal agencies, reduce costs, and improve citizen services. It is an initiative of the US Office of Management and Budget that aims to comply with the Clinger-Cohen Act.

Performance Reference Model (PRM)

The PRM is a standardized framework to measure the performance of major IT investments and their contribution to program performance.[1] The PRM has three main purposes:

1. Help produce enhanced performance information to improve strategic and daily decision-making;
2. Improve the alignment — and better articulate the contribution of — inputs to outputs and outcomes, thereby creating a clear “line of sight” to desired results; and



Performance reference model, 2005.[1]

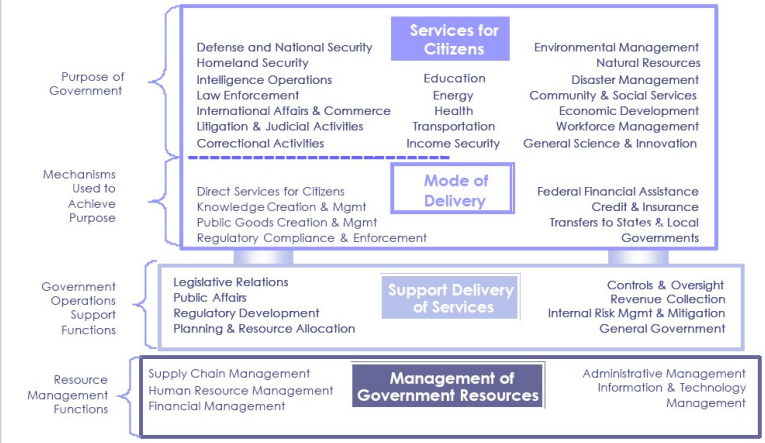
3. Identify performance improvement opportunities that span traditional organizational structures and boundaries

The PRM uses a number of existing approaches to performance measurement, including the Balanced Scorecard, Baldrige Criteria [5], value measuring methodology, program logic models, the value chain, and the Theory of Constraints. In addition, the PRM was informed by what agencies are currently measuring through PART assessments, GPRA, enterprise architecture, and Capital Planning and Investment Control. The PRM is currently composed of four measurement areas:

- Mission and Business Results
- Customer Results
- Processes and Activities
- Technology

Business Reference Model (BRM)

The "FEA business reference model" is a function-driven framework for describing the business operations of the Federal Government independent of the agencies that perform them. This business reference model provides an organized, hierarchical construct for describing the day-to-day business operations of the Federal government using a functionally driven approach. The BRM is the first layer of the Federal Enterprise Architecture and it is the main viewpoint for the analysis of data, service components and technology.[1]



Business Reference Model overview.[1]

The BRM is broken down into four areas:

- Services For Citizens
- Mode of Delivery
- Support Delivery of Services
- Management of Government Resources

The Business Reference Model provides a framework that facilitates a functional (as opposed to organizational) view of the federal government's LoBs, including its internal operations and its services for the citizens, independent of the agencies, bureaus and offices that perform them. By describing the federal government around common business areas instead of by a stovepiped, agency-by-agency view, the BRM promotes agency collaboration and serves as the underlying foundation for the FEA and E-Gov strategies.[1]

While the BRM does provide an improved way of thinking about government operations, it is only a model; its true utility can only be realized when it is effectively used. The functional approach promoted by the BRM will do little to help accomplish the goals of E-Government if it is not incorporated into EA business architectures and the management processes of all Federal agencies and OMB.[1]

Service Component Reference Model (SRM)

The Service Component Reference Model (SRM) is a business and performance-driven, functional framework that classifies Service Components with respect to how they support business and/or performance objectives.[1] The SRM is intended for use to support the discovery of government-wide business and application Service Components in IT investments and assets. The SRM is structured across horizontal and vertical service domains that, independent of the business functions, can provide a leverage-able foundation to support the reuse of applications, application capabilities, components, and business services.

Service Domains	Service Types
Customer Services	• Customer Relationship Management • Customer Preferences • Customer Initiated Assistance
Process Automation	• Tracking and Workflow • Routing and Scheduling
Business Management Services	• Management of Process • Organizational Management • Investment Management • Supply Chain Management
Digital Asset Services	• Content Management • Document Management • Knowledge Management • Records Management
Business Analytical Services	• Analysis and Statistics • Visualization • Knowledge Discovery • Business Intelligence • Reporting
Back Office Services	• Data Management • Human Resources • Financial Management • Asset / Materials Management • Development and Integration • Human Capital / Workforce Management
Support Services	• Security Management • Collaboration • Search • Communication • Systems Management • Forms Management

Service Component Reference Model.[6]

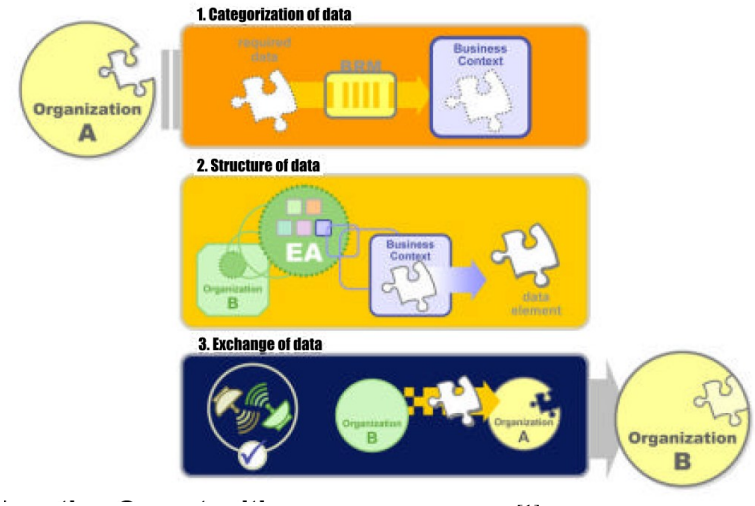
The SRM establishes the following domains:

- Customer Services
- Process Automation Services
- Business Management Services
- Digital Asset Services
- Business Analytical Services
- Back Office Services
- Support Services

Each Service Domain is decomposed into Service Types. For example, the three Service Types associated with the Customer Services Domain are: Customer Preferences; Customer Relationship Management; and Customer Initiated Assistance. And each Service Type is decomposed further into components. For example, the four components within the Customer Preferences Service Type include: Personalization; Subscriptions; Alerts and Notifications; and Profile Management.[6]

Data Reference Model (DRM)

The Data Reference Model (DRM) describes, at an aggregate level, the data and information that support government program and business line operations. This model enables agencies to describe the types of interaction and exchanges that occur between the Federal Government and citizens.[1] The DRM categorizes government information into greater levels of detail. It also establishes a classification for Federal data and identifies duplicative data resources. A common data model



The DRM Collaboration Process.[1]

will streamline information exchange processes within the Federal government and between government and external stakeholders.

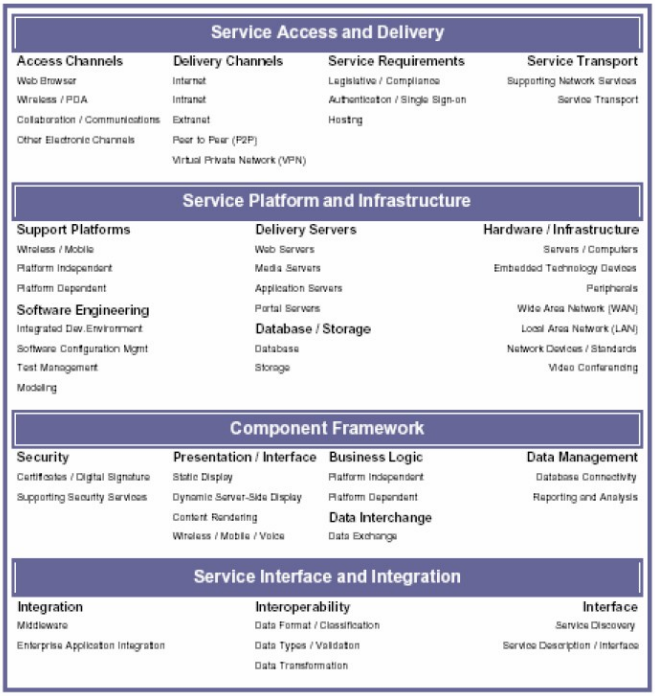
Volume One of the DRM provides a high-level overview of the structure, usage, and data-identification constructs. This document:

- Provides an introduction and high-level overview of the contents that will be detailed in Volumes 2–4 of the model;
- Encourages community of interest development of the remaining volumes; and
- Provides the basic concepts, strategy, and structure to be used in future development.

The DRM is the starting point from which data architects should develop modeling standards and concepts. The combined volumes of the DRM support data classification and enable horizontal and vertical information sharing.

Technical Reference Model (TRM)

The TRM is a component-driven, technical framework categorizing the standards and technologies to support and enable the delivery of Service Components and capabilities. It also unifies existing agency TRMs and E-Gov guidance by providing a foundation to advance the reuse and standardization of technology and Service Components from a government-wide perspective.[1]



Technical Reference Model.[1]

The TRM consists of:

- *Service Areas* : represent a technical tier supporting the secure construction, exchange, and delivery of Service Components. Each Service Area aggregates the standards and technologies into lower-level functional areas. Each Service Area consists of multiple Service Categories and Service Standards. This hierarchy provides the framework to group standards and technologies that directly support the Service Area. (Purple headings)
- *Service Categories* : classify lower levels of technologies and standards with respect to the business or technology function they serve. In turn, each Service Category comprises one or more Service Standards. (Bold-face groupings)
- *Service Standards* : define the standards and technologies that support a Service Category. To support agency mapping into the TRM, many of the Service Standards provide illustrative specifications or technologies as examples.(Plain text)

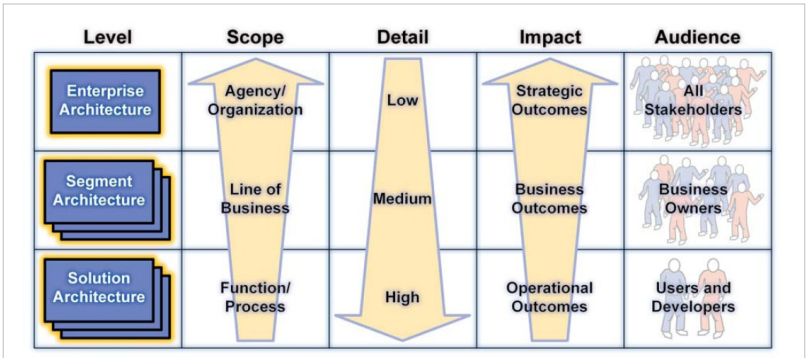
The figure on the right provides a high-level depiction of the TRM.

Aligning agency capital investments to the TRM leverages a common, standardized vocabulary, allowing interagency discovery, collaboration, and interoperability. Agencies and the federal government will benefit from economies of scale by identifying and reusing the best solutions and technologies to support their business functions, mission, and target architecture. Organized in a hierarchy, the TRM categorizes the standards and technologies that collectively support the secure delivery, exchange, and construction of business and application Service Components that may be used and leveraged in a component-based or service-oriented architecture.[1]

FEA Architecture levels

In the FEA enterprise, segment, and solution architecture provide different business perspectives by varying the level of detail and addressing related but distinct concerns. Just as enterprises are themselves hierarchically organized, so are the different views provided by each type of architecture. The Federal Enterprise Architecture Practice Guidance (2006) has defined three types of architecture:[3]

- Enterprise architecture,
- Segment architecture, and
- Solution architecture.



Federal Enterprise Architecture levels and attributes[3]

By definition, Enterprise Architecture (EA) is fundamentally concerned with identifying common or shared assets – whether they are strategies, business processes, investments, data, systems, or technologies. EA is driven by strategy; it helps an agency identify whether its resources are properly aligned to the agency mission and strategic goals and objectives. From an investment perspective, EA is used to drive decisions about the IT investment portfolio as a whole. Consequently, the primary stakeholders of the EA are the senior managers and executives tasked with ensuring the agency fulfills its mission as effectively and efficiently as possible.[3]

By contrast, "segment architecture" defines a simple roadmap for a core mission area, business service, or enterprise service. Segment architecture is driven by business management and delivers products that improve the delivery of services to citizens and agency staff. From an investment perspective, segment architecture drives decisions for a business case or group of business cases supporting a core mission area or common or shared service. The primary stakeholders for segment architecture are business owners and managers. Segment architecture is related to EA through three principles:

- structure: segment architecture inherits the framework used by the EA, although it may be extended and specialized to meet the specific needs of a core mission area or common or shared service.
- reuse : segment architecture reuses important assets defined at the enterprise level including: data; common business processes and investments; and applications and technologies.
- alignment : segment architecture aligns with elements defined at the enterprise level, such as business strategies, mandates, standards, and performance measures.[3]

"Solution architecture" defines agency IT assets such as applications or components used to automate and improve individual agency business functions. The scope of a solution architecture is typically limited to a single project and is used to implement all or part of a system or business solution. The primary stakeholders for solution architecture are system users and developers. Solution architecture is commonly related to segment architecture and enterprise architecture through definitions and constraints. For example, segment architecture provides definitions of data or service interfaces used within a core mission area or service, which are accessed by individual solutions. Equally, a solution may be constrained to specific technologies and standards that are defined at the enterprise level.[3]

FEA tools

A number of modeling tools enable you to capture the Federal Enterprise Architect reference models and align your enterprise architecture against them.

- Adaptive Inc.[7]
- Future Tech Systems, Inc.[8]
- IBM (formerly Telelogic) System Architect (software)
- alfabet with the software suite planningIT [9]
- Troux Technologies Architect
- Iteraplan – Open Source EA Tool
- Opentext Metastorm Provision [10]
- MEGA International MEGA Suite for Federal Enterprise Architecture [11]

The CIO Council's ET.gov site [12] can be used to identify technical specifications (standards) that are not yet included in the TRM but should be. Those that have been identified thus far can be discovered using the advanced ET.gov search service [13] hosted by IntelligenX [14].

References

[1] FEA Consolidated Reference Model Document. at whitehouse.gov May 2005. This document is revised to FEA Consolidated Reference Model Document Version 2.3 (http://www.whitehouse.gov/omb/assets/fea_docs/FEA_CRM_v23_Final_Oct_2007_Revised.pdf) October 2007. Accessed 28 April 2009.

[2] Chief Information Officer Council (2001) *A Practical Guide to Federal Enterprise Architecture* (http://www.enterprise-architecture.info/Images/Documents/Federal Enterprise Architecture Guide v1a.pdf). Feb 2001.

[3] Federal Enterprise Architecture Program Management Office (2007). FEA Practice Guidance (http://www.whitehouse.gov/sites/default/files/omb/assets/fea_docs/FEA_Practice_Guidance_Nov_2007.pdf).

[4] Overall the FEA is mandated by a series of federal laws and mandates. These federal laws have been:

- GPRA 1993 : Government Performance and Reform Act
- PRA 1995 : Paperwork Reduction Act
- CCA 1996 : Clinger-Cohen Act
- GPEA 1998 : The Government Paper work Elimination Act
- FISMA 2002 : Federal Information Security Management Act
- E-Gov 2002 : Electronic Government

Suplementairy OMB circulars have been:

- A-11 : Preparation, Submission and Execution of the Budget
- A-130 : OMB Circular A-130 Management of Federal Information Resources
- A-76 : Performance of Commercial Activities.

[5] http://www.nist.gov/baldrige/publications/criteria.cfm

[6] FEA (2005) FEA Records Management Profile, Version 1.0 (http://www.archives.gov/records-mgmt/pdf/rm-profile.pdf). December 15, 2005.

[7] (http://www.adaptive.com)

[8] Envision VIP (http://www.future-tech.com)

[9] http://www.alfabet.com/en/offering/approach/

[10] Metastorm (http://www.metastorm.com/)

[11] (http://www.mega.com/)

[12] http://et.gov

[13] http://etgov.i411.com/etgov/websearchservlet?toplevel=true

[14] http://www.intelligenx.com/

External links

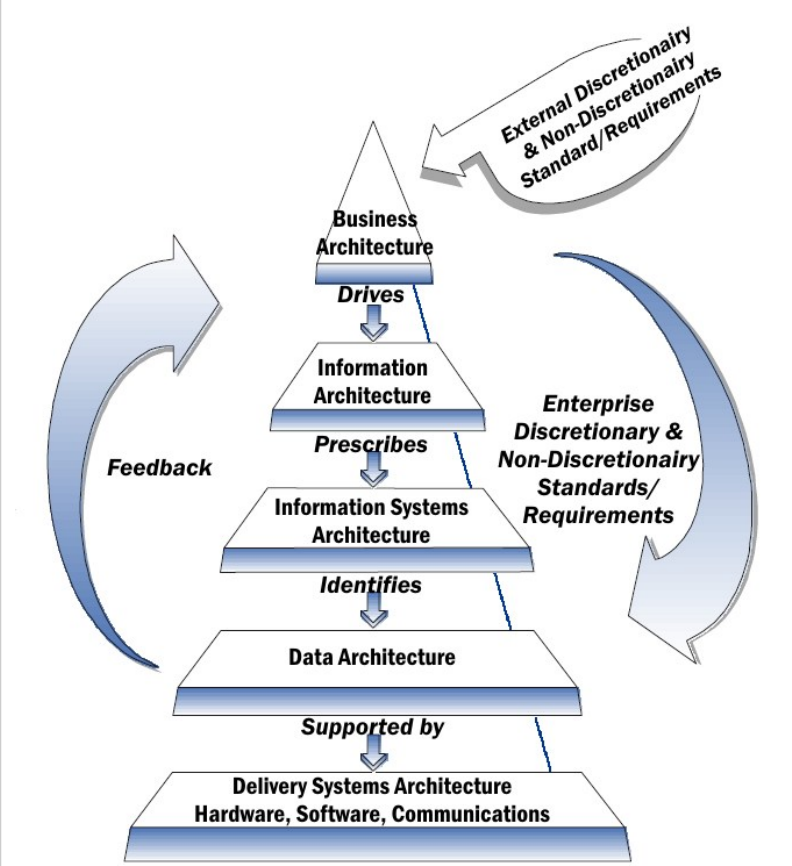
- e-gov FEA Program Office webpage (http://www.whitehouse.gov/omb/e-gov/fea/)
- Federal Enterprise Architecture Institute website (http://www.feacinstitute.org)
- Federal Chief Information Officers Council website (http://www.cio.gov)
- DoD CIO Enterprise Architecture & Standards (http://cio-nii.defense.gov/policy/eas.shtml)

NIST Enterprise Architecture Model

NIST Enterprise Architecture Model (**NIST EA Model**) is a reference model for Enterprise Architecture, that illustrates the interrelationship of enterprise business, information, and technology environments.[1]

This model developed by the National Institute of Standards and Technology (NIST) in 1989, became in the 1990s widely accepted and promoted within the U.S. federal government as an Enterprise Architecture management tool.[1]

This NIST Enterprise Architecture Model is the foundation of several U.S. federal Enterprise Architecture frameworks, for example the Federal Enterprise Architecture Framework.[1]



NIST Enterprise Architecture Model.[1]

Overview

The NIST Enterprise Architecture Model is a five-layered model allows for organizing, planning, and building an integrated set of information and information technology architectures. The five layers are defined separately but are interrelated and interwoven.[1] This interrelation between the architecture layers is defined in the model:[2]

- Business Architecture drives the information architecture
- Information architecture prescribes the information systems architecture
- Information systems architecture identifies the data architecture
- Data Architecture suggests specific data delivery systems, and
- Data Delivery Systems (Software, Hardware, Communications) support the data architecture.

The hierarchy in the model is based on the notion that an organization operates a number of business functions, each function requires information from a number of source, and each of these sources may operation one or more operation systems, which in turn contain data organized and stored in any number of data systems.[3]

History

The origin from the NIST Enterprise Architecture Model was a NIST research project in 1989, published as the NIST Special Publication 500-167, *Information Management Directions: The Integration Challenge*.[2] In this project two Frameworks were proposed: a Zachman Framework addressing enterprise engineering and a single dimensional classification of subject areas supporting Information Strategy, what latter became known as the NIST Framework. The NIST Framework was picked up by several U.S. federal agencies and used as the basis for their information strategy.[4]

NIST Enterprise Architecture topics

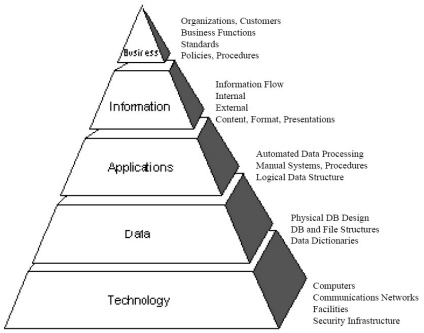
NIST

The U.S. National Institute of Standards and Technology (NIST) is responsible for developing technical, management, physical and administrative standards and guidelines; providing technical assistance; and conducting research for computer systems technology within the Federal government. Two NIST efforts in particular have provided direction to Federal agencies in the development of an open systems environment (OSE) to promote interoperability, portability, scalability, and standardization of agency information architectures and systems.[5]

- *Information Management Directions: The Integration Challenge*, NIST Special Publication 500-167, September 1989.
- *Application Portability Profile (APP) - The U.S. Government's Open System Environment Profile Version 3.0*, NIST Special Publication 500-230, February 1996.

Information Management Directions

The NIST report entitled "Information Management Directions: The Integration Challenge" defines the Enterprise Architecture, levels within the Architecture, and the standards required to implement and enforce such an Architecture. The NIST Enterprise Architecture has provided a framework for service and agency architecture model definitions. It consists of a five-tiered framework to illustrate business, information, and technology interconnectivity. Although the tiers are separately identified, they are interrelated. An integrated set of information and information technology architectures can be derived from the Enterprise Architecture. In the 1990s the Department of Energy (DOE) has adopted NIST's Enterprise Architecture model to define its Information Architecture, see image.[5]



DoE Information Architecture View.[5]

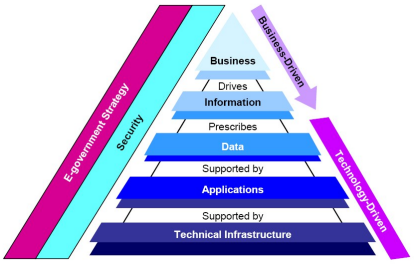
Application Portability Profile

The "Application Portability Profile (APP) - The U.S. Government's Open System Environment Profile Version 3.0" provides recommendations on a set of industry, Federal, national, international and other specifications that define interfaces, services, protocols, and data formats to support an Open System Environment (OSE). The APP addresses the lowest architecture in the NIST Enterprise Architecture Model, i.e., the Delivery System Architecture. Based on these specification recommendations, various services and agencies have defined detailed technical reference models. In the 1990s both the U.S. Patent and Trademark Office (PTO) of the Department of Commerce (DoC) and the Department of Defense (DoD) in its Technical Architecture Framework for Information Management (TAFIM) have defined their Technical Reference Models based on NIST's APP.[5]

Applications

NIST Enterprise Architecture Model is applicated the following frameworks:

- FDIC Enterprise Architecture Framework is the Enterprise Architecture framework of the Federal Deposit Insurance Corporation (FDIC).
- FEAF : The 1999 documentation of the Federal Enterprise Architecture Framework Version 1.1 explains how the NIST Framework is used as a foundation of the FEA Framework.[1]
- NWS Enterprise Architecture : Enterprise Architecture of the National Weather Service[7]



FDIC's Enterprise Architecture Framework.[6]

References

This article incorporates public domain material from websites or documents of the National Institute of Standards and Technology [7].

[1] The Chief Information Officers Council (1999). Federal Enterprise Architecture Framework Version 1.1 (http://www.cio.gov/documents/fedarch1.pdf). September 1999.

[2] Elizabeth N. Fong and Alan H. Goldfine (1989) *Information Management Directions: The Integration Challenge*. National Institute of Standards and Technology (NIST) Special Publication 500-167, September 1989.

[3] John O'Looney (2002). *Wiring Governments: Challenges and Possibilities for Public Managers*. Greenwood Publishing Group. p.67.

[4] "Exclusive Interview with John Zachman" (http://www.objectwatch.com/whitepapers/IASANewsletterApril2007.pdf) by Roger Sessions. In: *Perspectives of the International Association of Software Architects*. April 2006.

[5] Federal Aviation Administration (1998) Federal Information Architecture Initiatives (http://www.faa.gov/niac/pdf/wn18_fia.pdf). February 1998

[6] OIG (2005). Implementation of E-Government Principles (http://www.fdicoig.gov/reports05/05-018-508.shtml). May 2005

[7] Bobby Jones (2003) NWS ENTERPRISE ARCHITECTURE (http://ams.confex.com/ams/pdfpapers/73211.pdf).

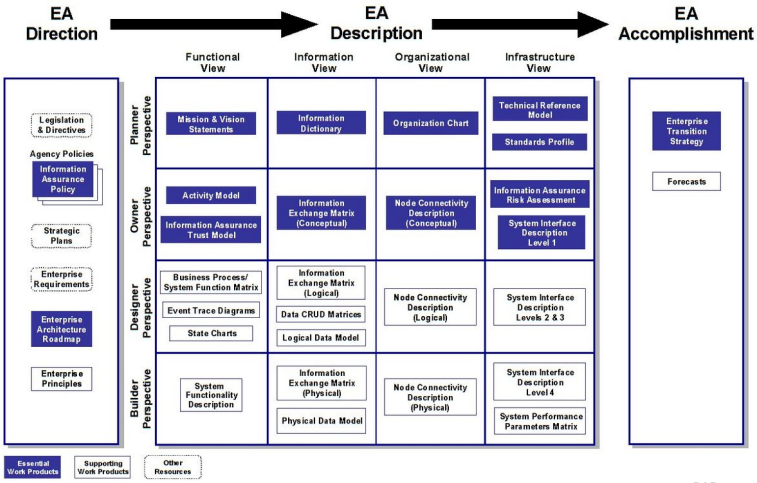
External links

- Federal Information Architecture Initiatives (http://www.faa.gov/niac/pdf/wn18_fia.pdf) in the 1990s.

Treasury Enterprise Architecture Framework

Treasury Enterprise Architecture Framework (TEAF) is an Enterprise architecture framework for treasury, based on the Zachman Framework.

It was developed by the US Department of the Treasury and published in July 2000.[2]



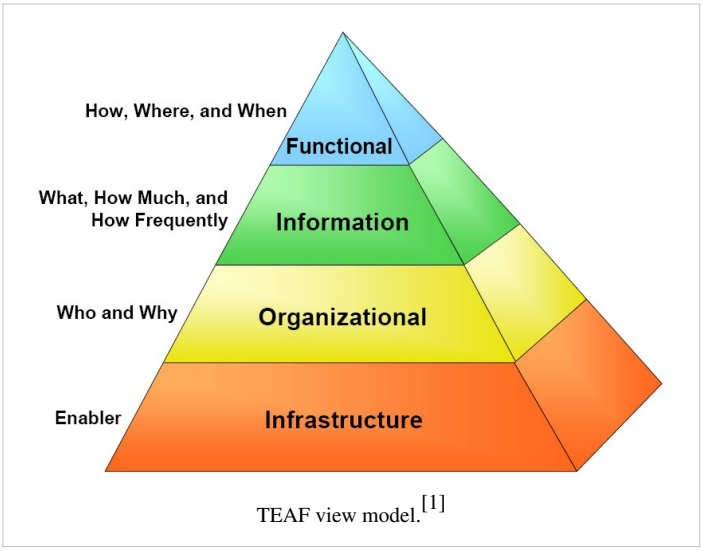
TEAF Work Products for EA Direction, Description, and Accomplishment.[1]

Overview

The Treasury Enterprise Architecture Framework (TEAF) an architectural framework that supports Treasury's business processes in terms of products. This framework guides the development and redesign of the business processes for various bureaus in order to meet the requirements of recent legislation in a rapidly changing technology environment. The TEAF prescribes architectural views and delineates a set of notional products to portray these views.[1]

The TEAF describes provides:[1]

- Guidance to Treasury bureaus concerning the development and evolution of information systems architecture,
- A unifying concept, common principles, technologies, and standards for information systems, and
- A template for the development of the Enterprise Architecture.

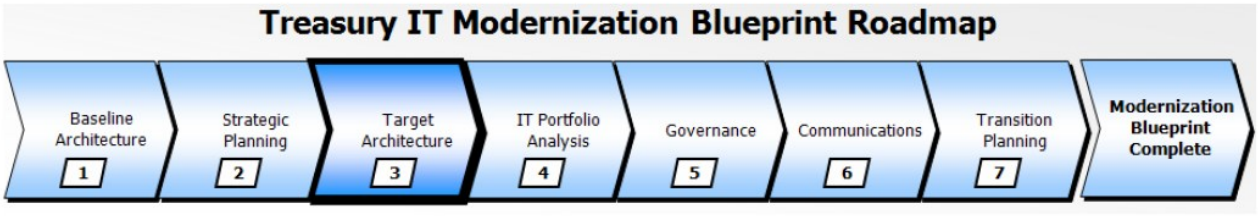


TEAF view model.[1]

The TEAF's functional, information and organizational architecture views collectively model the organization's processes, procedures, and business operations. By grounding the architecture in the business of the organization, the TEAF defines the core business procedures and enterprise processes. Through its explicit models, a TEAF-based architecture enables the identification and reasoning of enterprise- and system-level concerns and investment decisions.[1]

History

The Treasury Enterprise Architecture Framework (TEAF) is derived from earlier treasury models, such as the US Treasury model (TISAF) released 1997, and the Federal Enterprise Architecture Framework (FEAF), released in 1999.[3] The first version of the TEAF was released July 2000.

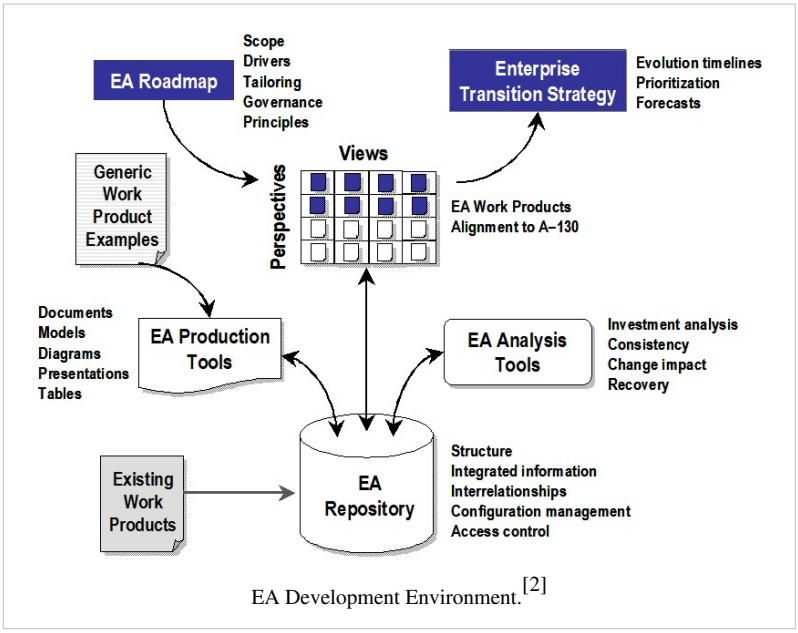


Blueprint Roadmap to Treasury IT Modernization.[4]

In the new millennium the Treasury Enterprise Architecture Framework (TEAF) has developed into the Treasury Enterprise Architecture (TEA), which aims to establish a roadmap for the modernization and optimization of the U.S. Treasury Department's business processes and IT environment. The Treasury Enterprise Architecture will provide a framework to guide IT investment planning, streamline systems, and ensure that IT programs align with business requirements and strategic goals.[5]

TEAF Topics

Enterprise Architecture



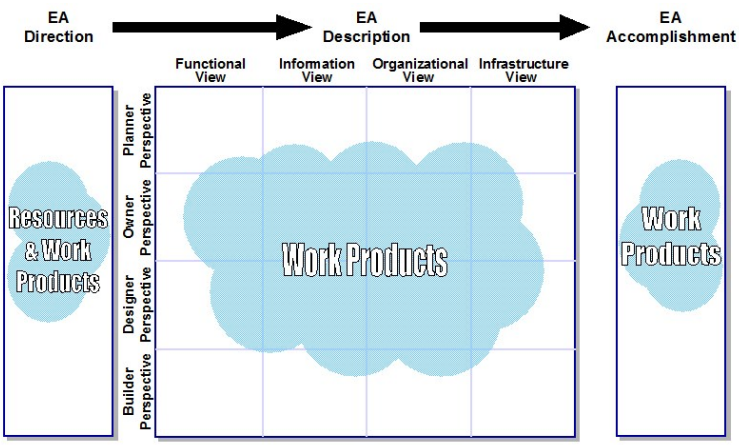
EA Development Environment.[2]

Effective management and strategic decision making, especially for information technology (IT) investments, require an integrated view of the enterprise—understanding the interrelationships among the business organizations, their operational processes, and the information systems that support them. An Enterprise Architecture formalizes the identification, documentation, and management of these interrelationships, and supports the management and decision processes. The Enterprise Architecture provides substantial support for evolution of an enterprise as it anticipates and responds to the changing needs of its customers and constituents. The Enterprise Architecture is a vital part of the enterprise's decision-making process, and will evolve along with the enterprise's mission.[2]

The TEAF has been designed to help both the bureaus and the Department develop and maintain their Enterprise Architectures. The TEAF aims to establish a common Enterprise Architecture structure, consistent practices, and common terminology; and to institutionalize Enterprise Architecture governance across the Department. This architectural consistency will facilitate integration, information sharing, and exploitation of common requirements across Treasury.[2]

Enterprise Architecture Framework

The purpose of the Enterprise Architecture Framework is to provide a structure for producing an Enterprise Architecture (EA) and managing Enterprise Architecture assets. To reduce the complexity and scope of developing and using an Enterprise Architecture, it must be subdivided so that portions may be used independently or built incrementally in separate projects. The TEAF subdivides an Enterprise Architecture by:[2]



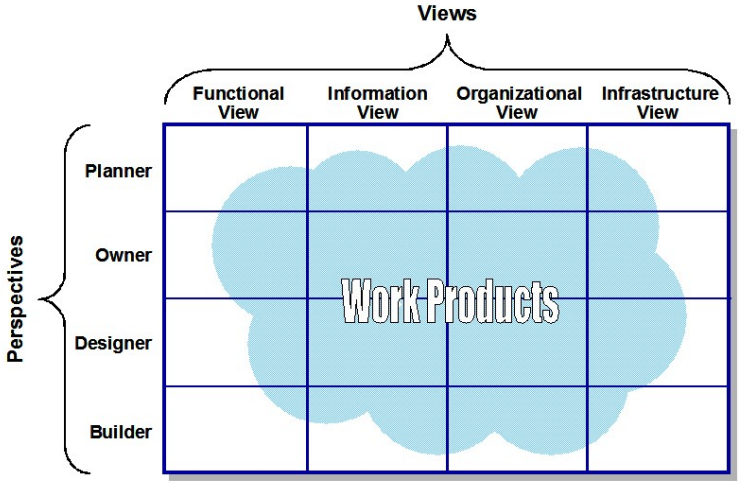
Overview of a Framework for EA Direction, Description, and Accomplishment Overview.[2]

- Views
- Perspectives
- Work products

The TEAF identifies, as shown in the figure, resources and work products that provide direction for EA development, work products constituting the EA description, and work products documenting how to accomplishment an EA implementation. The resources and work products for EA direction and accomplishment are not part of the EA description itself, but are developed and applied during the overall enterprise life cycle. The TEAF Matri, organizes the subdivisions of the EA description and demonstrates the relationships among them. The following sections describe the subdivisions of the EA and their relationships to the TEAF Matrix.[2]

TEAF Matrix of Views and Perspectives

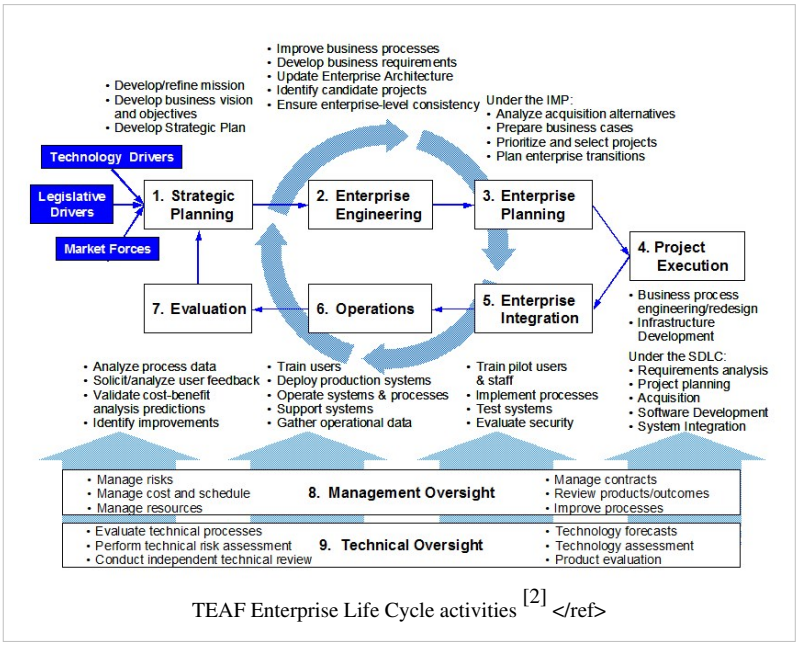
The TEAF Matrix is a simplified portrayal of an EA structure to aid in understanding important EA aspects from various vantage points (views and perspectives). The TEAF Matrix aims to provide a simple, uniform structure to an entire framework. As depicted in the figure, the TEAF Matrix consists of four architectural views (Functional, Information, Organizational, and Infrastructure), which are shown as columns, and four perspectives (Planner, Owner, Designer, and Builder), which appear as rows. The TEAF Matrix is a four-by-four matrix with a total of 16 cells. The views and perspectives are described in the following sections.[2]



TEAF Matrix of Views and Perspectives.[2]

When an EA description work product is shown within one cell of the TEAF Matrix, it means that the main vantage points for developing that work product correspond to that column (view) and row (perspective). However, information from other views (and sometimes other perspectives) is needed to produce a work product. Not all cells must be “filled-in” by producing an associated work product. Each bureau must define in its EA Roadmap its plans for producing and using an EA to match its needs.[2]

Enterprise Life Cycle activities



TEAF Enterprise Life Cycle activities [2] </ref>

An Enterprise Life Cycle integrates the management, business, and engineering life cycle processes that span the enterprise to align its business and IT activities. Enterprise Life Cycle refers generally to an organization's approach for managing activities and making decisions during ongoing refreshment of business and technical practices to support its enterprise mission. These activities include investment management, project definition, configuration management, accountability, and guidance for systems development according to a System Development Life Cycle (SDLC). The Enterprise Life Cycle applies to enterprise-wide planning activities and decision making. By contrast, a System Development Life Cycle generally refers to practices for building individual systems. Determining what systems to build is an enterprise-level decision.[2]

The figure on the left depicts notional activities of an Enterprise Life Cycle methodology. Within the context of this document, Enterprise Life Cycle does not refer to a specific methodology or a specific bureau's approach. Each organization needs to follow a documented Enterprise Life Cycle methodology appropriate to its size, the complexity of its enterprise, and the scope of its needs.[2]

Products

The TEAF provides a unifying concept, common terminology and principles, common standards and formats, a normalized context for strategic planning and budget formulation, and a universal approach for resolving policy and management issues. It describes the enterprise information systems architecture and its components, including the architecture's purpose, benefits, characteristics, and structure. The TEAF introduces various architectural views and delineates several modeling techniques. Each view is supported with graphics, data repositories, matrices, or reports (i.e., architectural products).[1]

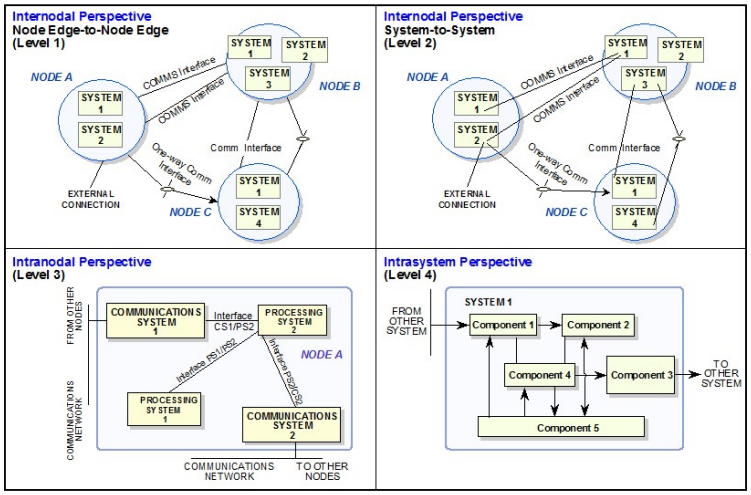
	Functional View	Information View	Organizational View	Infrastructure View	
Planner Perspective	Mission & Vision Statements	Information Dictionary	Organization Charts	Technical Reference Model Standards Profile	EA Repository Listings
Owner Perspective	Activity Model Info Assurance Trust Model	Information Exchange Matrix (Conceptual)	Node Connectivity Description (Conceptual)	Info Assurance Risk Assessment System Interface Description (Level 1)	High Level Modeling
Designer Perspective	Business Process/ System Function Matrix Event Trace Diagrams State Charts	Information Exchange Matrix (Logical) Logical Data Model Data CRUD Matrices	Node Connectivity Description (Logical)	System Interface Description (Level 2 & 3)	Logical Modeling
Builder Perspective	System Functionality Description	Information Exchange Matrix (Physical) Physical Data Model	Node Connectivity Description (Physical)	System Interface Description (Level 4) System Performance Parameters Matrix	Physical Modeling
	Essential Work Products	Supporting Work Products			

TEAF Products.[1]

The figure shows a matrix with four views and four perspectives. Essential products are shown across the top two rows of the matrix. It is notable that the TEAF includes an Information Assurance Trust model, the Technical Reference Model, and standards profiles as essential work products. These are not often addressed as critical

framework components. One of these frameworks should provide a means to logically structure and organize the selected EA products. Now, in order to effectively create and maintain the EA products, a toolset should be selected.[1]

System Interface Description



System Interface Description, Levels 1, 2, 3, 4—Generic Examples.

The System Interface Description (SID) links together the Organizational and Infrastructure Views by depicting the assignments of systems and their interfaces to the nodes and needlines described in the Node Connectivity Description. The Node Connectivity Description for a given architecture shows nodes (not always defined in physical terms), while the System Interface Description depicts the systems corresponding to the system nodes. The System Interface Description can be produced at four levels, as described below. Level 1 is an essential work product, while Levels 2, 3, and 4 are supporting work products.[2]

The System Interface Description identifies the interfaces between nodes, between systems, and between the components of a system, depending on the needs of a particular architecture. A system interface is a simplified or generalized representation of a communications pathway or network, usually depicted graphically as a straight line, with a descriptive label. Often, pairs of connected systems or system components have multiple interfaces between them. The System Interface Description depicts all interfaces between systems and/or system components that are of interest to the architect.[2]

The graphic descriptions and/or supporting text for the System Interface Description should provide details concerning the capabilities of each system. For example, descriptions of information systems should include details concerning the applications present within the system, the infrastructure services that support the applications, and the means by which the system processes, manipulates, stores, and exchanges data.[2]

References

[1] FEA Consolidated Reference Model Document (http://georgewbush-whitehouse.archives.gov/omb/egov/documents/CRM.PDF). whitehouse.gov May 2005.

[2] US Department of the Treasury Chief Information Officer Council (2000). Treasury Enterprise Architecture Framework (http://www.eaframeworks.com/TEAF/teaf.doc). Version 1, July 2000.

[3] Jaap Schekkerman (2003). *How to Survive in the Jungle of Enterprise Architecture Frameworks*. p.113

[4] Standards and Configuration Management Team (SCMT) of the Treasury Enterprise Architecture Sub-Council (TEAC) (2007). Treasury Technical Standards Profile (http://www.treas.gov/offices/cio/egov/ea/Technical_Standards_Profile.pdf). May 2007

[5] E-Government (http://www.treas.gov/offices/cio/egov/) U.S. Treasury Department, accessed 09 Dec 2008.

External links

- U.S. Treasury - Office of the CIO (http://www.treas.gov/offices/cio/) homepage.
- Other Architectures and Frameworks (http://www.opengroup.org/architecture/togaf8-doc/arch/chap37.html), The Open Group 1999-2006.

Lifecycles

Enterprise life cycle

Enterprise Life Cycle (ELC) in enterprise architecture is the dynamic, iterative process of changing the enterprise over time by incorporating new business processes, new technology, and new capabilities, as well as maintenance, disposition and disposal of existing elements of the enterprise.[1]

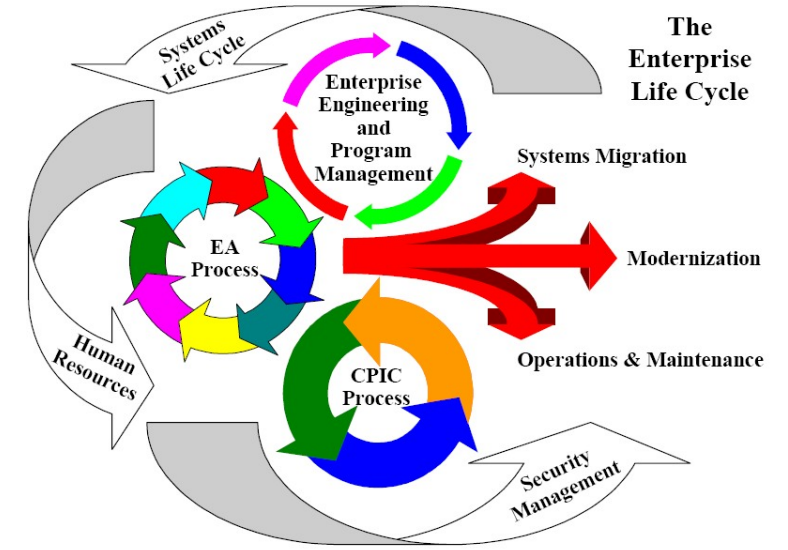


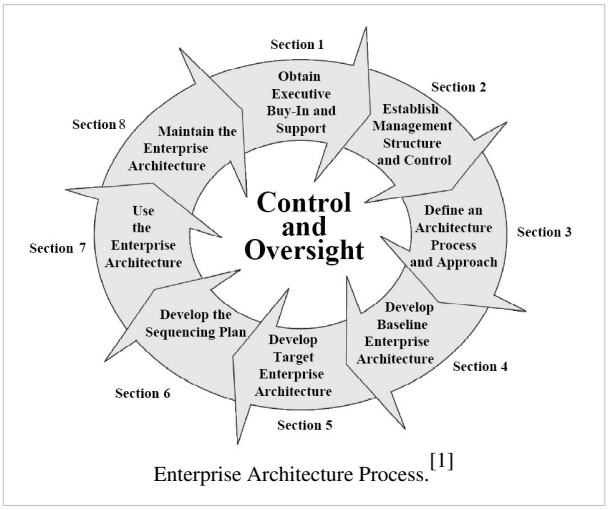
Illustration of the Enterprise Life Cycle.[1]

Overview

The enterprise life cycle is a concept in Enterprise Architecture (EA). The Enterprise Architecture process is closely related to other processes, such as enterprise engineering and program management cycle, more commonly known as the Systems Development Life Cycle. This concept aids in the implementation of an Enterprise Architecture, and the Capital Planning and Investment Control (CPIC) process that selects, controls, and evaluates investments. Overlying these processes are human capital management and information security management. When these processes work together effectively, the enterprise can effectively manage information technology as a strategic resource and business process enabler. When these processes are properly synchronized, systems migrate efficiently from legacy technology environments through evolutionary and incremental developments, and the Agency is able to demonstrate its return on investment (ROI). The figure on top illustrates the interaction of the dynamic and interactive cycles as they would occur over time.[1]

Enterprise Life Cycle topics

Enterprise Architecture Process

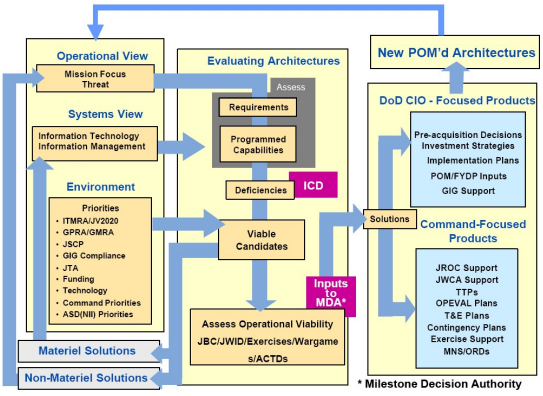


Enterprise Architecture Process.[1]

As a prerequisite to the development of every enterprise architecture, each Agency should establish the need to develop an EA and formulate a strategy that includes the definition of a vision, objectives, and principles. The figure shows a representation of the EA process. Executive buy-in and support should be established and an architectural team created within the organization. The team defines an approach and process tailored to Agency needs. The architecture team implements the process to build both the baseline and target EAs. The architecture team also generates a sequencing plan for the transition of systems, applications, and associated business practices predicated upon a detailed gap analysis. The architecture is employed in the CPIC and the enterprise engineering and program management processes via prioritized, incremental projects and the insertion of emerging new technologies. Lastly, the architectures are maintained through a continuous modification to reflect the Agency's current baseline and target business practices, organizational goals, visions, technology, and infrastructure.[1]

Architecture Life Cycle

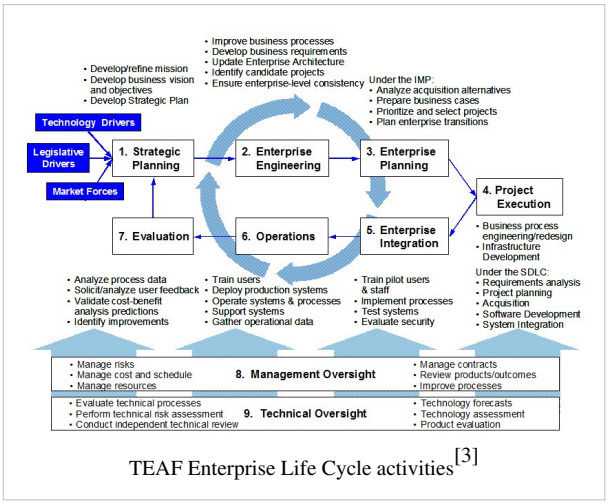
The figure depicts the life of the architecture as it evolves and shows the process that the architecture description supports in the development, analysis, and evolution of the implemented architecture. In this illustration, the Operational View is used to drive the requirements that are evaluated against the Systems View. Operational deficiencies are derived from the analysis, and viable candidates are identified. These candidates can take the form of either materiel or non- materiel solutions and are modeled back into the Operational and Systems Views of the architecture.[2]



DoDAF Architecture Life Cycle.[2]

The architecture is re-analyzed, and the process continues until the operational deficiencies are minimized. The final sets of viable candidates are assessed for operational viability. Based on the results of the assessments, design changes are made and submitted for inclusion into the budgeting process. This process of developing, analyzing, and modifying continues throughout the architecture's life cycle.[2]

Enterprise Life Cycle activities



TEAF Enterprise Life Cycle activities[3]

An Enterprise Life Cycle integrates the management, business, and engineering life cycle processes that span the enterprise to align its business and IT activities. Enterprise Life Cycle refers generally to an organization's approach for managing activities and making decisions during ongoing refreshment of business and technical practices to support its enterprise mission. These activities include investment management, project definition, configuration management, accountability, and guidance for systems development according to a System Development Life Cycle (SDLC). The Enterprise Life Cycle applies to enterprise-wide planning activities and decision making. By contrast, a System Development Life Cycle generally refers to practices for building individual systems. Determining what systems to build is an enterprise-level decision.[3]

The figure on the right depicts notional activities of an Enterprise Life Cycle methodology. Within the context of this document, Enterprise Life Cycle does not refer to a specific methodology or a specific bureau's approach. Each organization needs to follow a documented Enterprise Life Cycle methodology appropriate to its size, the complexity of its enterprise, and the scope of its needs.[3]

Enterprise Performance Life Cycle

The Enterprise Performance Life Cycle (EPLC) encompasses the major business functions executed under the Office of the Chief Information Officer (CIO), and in particular shows at a high level the relationship among the different business functions and both the general order and the iterative nature of their execution. The placement of enterprise architecture in the center of the EPLC conceptual diagram, shown in the figure, reflects the supporting and enabling role that enterprise architecture serves for the major business functions in the Enterprise Performance Life Cycle.[4]

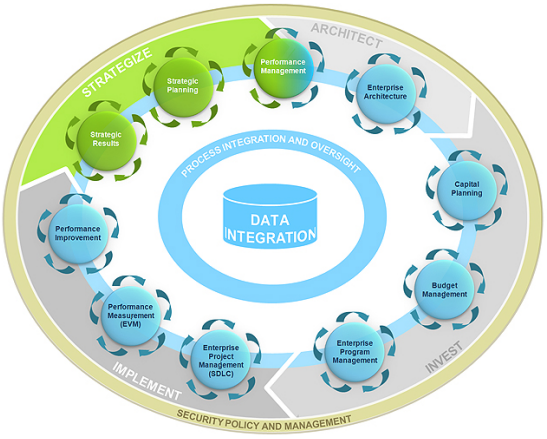


Illustration of the Enterprise Performance Life Cycle of the U.S. Department of Health & Human Services.[4]

The Enterprise Architecture (EA) Program explicitly considers the information needs of the Enterprise Performance Life Cycle (EPLC) processes in developing and enhancing the EA Framework, collecting and populating data in the EA Repository, and developing views, reports, and analytical tools that can be used to facilitate the execution of the EPLC processes. The EPLC conceptual diagram in the figure provides a Departmental perspective of key business functions. The EPLC is also relevant from an individual investment or project perspective, as each new investment passes through each phase of the EPLC. The investment-level perspective is detailed in the Enterprise Performance Life Cycle Framework.[4]

References

[1] Chief Information Officer Council (2001). A Practical Guide to Federal Enterprise Architecture (http://www.gao.gov/bestpractices/bpeaguide.pdf)

[2] DoD Architecture Framework Working Group (2003). DoD Architecture Framework Version 1.0 Deskbook (https://acc.dau.mil/CommunityBrowser.aspx?id=31667&lang=en-US) 15 August 2003.

[3] US Department of the Treasury Chief Information Officer Council (2000). Treasury Enterprise Architecture Framework (http://www.eaframeworks.com/TEAF/teaf.doc). Version 1, July 2000.

[4] U.S. Department of Health & Human Services (2007). HHS Enterprise Architecture Governance Plan FY2007 (http://www.hhs.gov/ocio/ea/documents/hhs_ea_governance_plan_ocio.html).

Further reading

- Alain Bernard, Serge Tichkiewitch (2008). *Methods and Tools for Effective Knowledge Life-Cycle-Management*.
- Peter Bernus, Laszlo Nemes, Günter Schmidt (2003). *Handbook on Enterprise Architecture*.
- Jeffrey O. Grady (2006). *System requirements analysis*
- Arturo Molina, Jose Manuel Sanchez, Andrew Kusiak (1998). *Handbook of Life Cycle Engineering: Concepts, Models, and Technologies*.
- François Vernadat (1996). *Enterprise Modeling and Integration: Principles and Applications*.

External links

- Enterprise Life Cycle Management (http://www.epa.gov/oei/symposium/2005/sullivan.pdf) presentation 2005
- EA in the Federal Enterprise Life Cycle EA in the Federal Enterprise Life Cycle (http://colab.cim3.net/file/work/BPC/2006-09-07/Sept07-04_Follow-the-Money.ppt) presentation 2006.

ISO 12207

ISO/IEC 12207 *Systems and software engineering — Software life cycle processes*[1] is an international standard for software lifecycle processes. It aims to be *the* standard that defines all the tasks required for developing and maintaining software.

The ISO/IEC 12207 standard establishes a process of lifecycle for software, including processes and activities applied during the acquisition and configuration of the services of the system. Each Process has a set of outcomes associated with it. There are 23 Processes, 95 Activities, 325 Tasks and 224 Outcomes (the new "ISO/IEC 12207:2008 Systems and software engineering – Software life cycle processes" defines 43 system and software processes).

The standard has the main objective of supplying a common structure so that the buyers, suppliers, developers, maintainers, operators, managers and technicians involved with the software development use a common language. This common language is established in the form of well defined processes. The structure of the standard was intended to be conceived in a flexible, modular way so as to be adaptable to the necessities of whoever uses it. The standard is based on two basic principles: modularity and responsibility. Modularity means processes with minimum coupling and maximum cohesion. Responsibility means to establish a responsibility for each process, facilitating the application of the standard in projects where many people can be legally involved.

The set of processes, activities and tasks can be adapted according to the software project. These processes are classified in three types: basic, for support and organizational. The support and organizational processes must exist independently of the organization and the project being executed. The basic processes are instantiated according to the situation.

Primary lifecycle processes

The primary lifecycle processes contain the core processes involved in creating a software product. These processes are divided into five different main processes:

- Acquisition
- Supply
- Development
- Operation
- Maintenance
- Destruction

Because the primary lifecycle processes cover a very large area a scope was defined. This entry explains all the primary lifecycle processes but will explain the Acquisition and Development processes more extensively.

Activities

Each phase within the primary lifecycle processes can be divided into different activities. This chapter explains the different activities for each primary lifecycle process..

Acquisition

Acquisition covers the activities involved in initiating a project. The acquisition phase can be divided into different activities and deliverables that are completed chronologically.

- Initiation: during this activity the following tasks are completed
  - The need is described why to acquire, develop, or enhance a product;
  - System requirements are defined and approved if applicable;
  - The global software requirements are defined;
  - Evaluation of other options, like a purchase of an off-the-shelf product or enhancement of an existing product;
  - If an off-the-shelf product is purchased, the software requirements of this product need to be analyzed.
  - An acquisition plan is developed, this plan will be used further on during the acquisition phase
  - Acceptance criteria are defined.
- Request for proposal preparation: during this activity the following tasks are completed
  - Acquisition requirements, like System requirements and technical constraints such as target environment, are defined.
  - Required ISO/IEC 12207 process for the project are defined and changed accordingly if needed.
  - Contract milestones for reviewing and supplier's progress audits are defined.
- Prepare Contract: during this activity the following tasks are completed
  - Selection procedure for suppliers are developed;
  - Suppliers, based on the developed selection procedure, are selected;
  - The tailor-made ISO/IEC 12207 standard must be included in the contract;
- Negotiate changes: during this activity the following tasks are completed
  - Negotiations are held with the selected suppliers;
- Update contract: during this activity the following tasks are completed
  - Contract is updated with the result from the negotiations in the previous activity.
- Supplier monitoring: during this activity the following tasks are completed
  - Activities of the suppliers according to the agreements made are monitored;
  - Work together with suppliers to guarantee timely delivery if needed.
- Acceptance and completion: during this activity the following tasks are completed

- Acceptance tests and procedures are developed;
- Acceptance and testing on the product is conducted;
- Configuration management on the delivered product is conducted;

Supply

During the supply phase a project management plan is developed. This plan contains information about the project such as different milestones that need to be reached. This project management plan is needed during the next phase which is the development phase.

Development

During the development phase the software product is designed, created and tested and will result in a software product ready to be released to the customer. Throughout time many people have developed means of developing a software application. The choice of developing method often depends on the present situation. The development method which is used in many projects is the V-model. Techniques that can be used during the development are UML for designing and TMap for testing. This entry contains the most important steps of the V-model.

- Define functional requirements: during this activity the following tasks are completed
  - Gather the functional requirements, or demands, for the product that is to be created.
- Create High level design: during this activity the following tasks are completed
  - A basic layout of the product is created. This means the setup of different modules and how they communicate with each other. This design does not contain very much detail about the modules.
- Create Module design
  - The different modules present in the High level design are designed separately. The modules are designed in as much detail as possible.
- Coding
  - The code is created according to the high level design and the module design.
- Execute Module test
  - The different modules are tested for correct functioning. If this is the case the project can move to the next activity, else the project returns to the module design phase to correct any errors.
- Execute Integration test
  - The communication between modules is tested for correct functioning. If this is the case the project can move to the next activity, else the project falls back to the high level design to correct any errors.
- Execute System test
  - This test checks whether all functional requirements are present in the product. If this is the case the product is completed and the product is ready to be transferred to the customer. Else the project falls back to the software requirements activity and the functional requirements have to be adjusted.

Operation

The operation and maintenance phases occur simultaneously, the operation-phase consists of activities like assisting users in working with the created software product.

Maintenance

The maintenance-phase consists of maintenance-tasks to keep the product up and running. The maintenance includes any general enhancements, changes and additions, which might be required by the end-users.These defects and deficiencies are usually documented by the developing organisation to enable future solutions and known issues addressing in any future maintenance releases. There is no disposal phase

Deliverables

The different deliverables that are developed per activity are explained in this chapter.

Acquisition

Acquisition covers the activities involved in initiating a project. The acquisition phase can be divided into different activities and deliverables that are completed chronologically.

- Initiation: during this activity the following deliverables are developed:
  - Initiation documents;
- Request for proposal preparation: during this activity the following deliverables are developed:
  - Request for proposal;
- Prepare Contract: during this activity the following deliverables are developed:
  - Contract: this is a draft agreement between the company and suppliers, set up by the company.
- Negotiate Changes: during this activity the following deliverables are developed:
  - Input from the suppliers: suppliers can react on the draft agreement submitted by the company, this reaction will result in input from the suppliers
- Update Contract: during this activity the following deliverables are developed:
  - Final Contract;
- Supplier monitoring: during this activity the following deliverables are developed:
  - Supplier Monitor Report: this report covers the advances of the suppliers involved based on different milestones.
- Acceptance and completion: during this activity the following deliverables are developed:
  - Acquisition report: this report covers the acceptance and completion of the acquisition phase.

Development

During the development phase the software product is designed, created and tested and will result in a software product ready to be sold to the customer.

- Define Software Requirements: during this activity the following deliverables are developed:
  - Software Requirements: this is a collection of different functional requirements;
- High level design: during this activity the following deliverables are developed:
  - High level design;
- Module design: during this activity the following deliverables are developed:
  - Module design;
- Coding: during this activity the following deliverables are developed:
  - Code;
- Module test: during this activity the following deliverables are developed:
  - Module test report, this test report contains the test-results which are formed after a module test of the application. Based on this test-report the project-team can decide which action to undertake further.
- Integration test: during this activity the following deliverables are developed:
  - Integration test report, this test report contains the test-results which are formed after an integration test of the application. Based on this test-report the project-team can decide which action to undertake further.
- System test: during this activity the following deliverables are developed:
  - System test report;

Example

The method presented in this entry can be used in a company that is responsible for creating and maintaining a software product for a customer. Especially when this company decides to build an application from scratch and that maintenance and assisting in the operation is also done by the company developer.

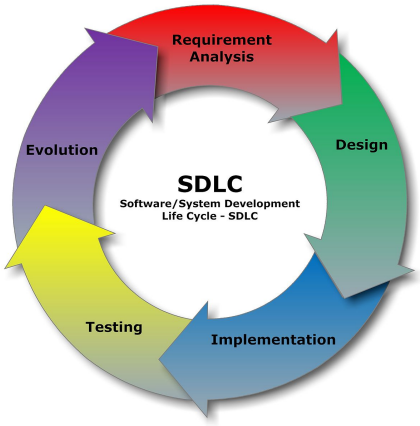
References

[1] ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes* (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43447)

- Mitchell H. Levine (2006). "Analyzing the Deliverables Produced in the Software Development Life Cycle" (http://www.auditserve.com/Articles/AuditGeneral/AnalyzingthedeliverablesproducedintheSDLC/tabid/250/Default.aspx). Audit Serve, Inc. Retrieved 2011-10-28.
- "SSC San Diego Process Asset Library" (http://sepo.spawar.navy.mil/12207.doc). Retrieved 2006-02-19.

Systems Development Life Cycle

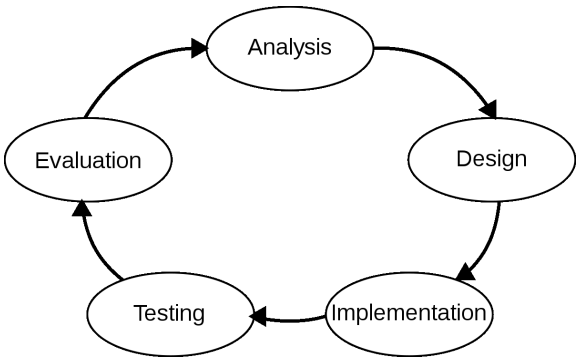
The **Systems development life cycle (SDLC)**, or **Software development process** in systems engineering, information systems and software engineering, is a process of creating or altering information systems, and the models and methodologies that people use to develop these systems. In software engineering, the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system[1]: the software development process.



Model of the Systems Development Life Cycle

Overview

The SDLC is a process used by a systems analyst to develop an information system, training, and user (stakeholder) ownership. Any SDLC should result in a high quality system that meets or exceeds customer expectations, reaches completion within time and cost estimates, works effectively and efficiently in the current and planned Information Technology infrastructure, and is inexpensive to maintain and cost-effective to enhance.[2] Computer systems are complex and often (especially with the recent rise of service-oriented architecture) link multiple traditional systems potentially supplied by different software vendors. To manage this level of complexity, a number of SDLC models or methodologies have been created, such as "waterfall"; "spiral"; "Agile software development"; "rapid prototyping"; "incremental"; and "synchronize and stabilize".[3]



Model of the Systems Development Life Cycle

SDLC models can be described along spectrum of agile to iterative to sequential. Agile methodologies, such as XP and Scrum, focus on lightweight processes which allow for rapid changes along the development cycle. Iterative methodologies, such as Rational Unified Process and dynamic systems development method, focus on limited

project scope and expanding or improving products by multiple iterations. Sequential or big-design-up-front (BDUF) models, such as Waterfall, focus on complete and correct planning to guide large projects and risks to successful and predictable results . Other models, such as Anamorphic Development, tend to focus on a form of development that is guided by project scope and adaptive iterations of feature development.

In project management a project can be defined both with a project life cycle (PLC) and an SDLC, during which slightly different activities occur. According to Taylor (2004) "the project life cycle encompasses all the activities of the project, while the systems development life cycle focuses on realizing the product requirements".[4] SDLC (systems development life cycle) is used during the development of an IT project, it describes the different stages involved in the project from the drawing board, through the completion of the project.

History

The **systems life cycle (SLC)** is a methodology used to describe the process for building information systems, intended to develop information systems in a very **deliberate**, structured and methodical way, reiterating each stage of the life cycle. The systems development life cycle, according to Elliott & Strachan & Radford (2004), "originated in the 1960's,to develop large scale functional business systems in an age of large scale business conglomerates. Information systems activities revolved around heavy data processing and number crunching routines".[5]

Several systems development frameworks have been partly based on SDLC, such as the structured systems analysis and design method (SSADM) produced for the UK government Office of Government Commerce in the 1980s. Ever since, according to Elliott (2004), "the traditional life cycle approaches to systems development have been increasingly replaced with alternative approaches and frameworks, which attempted to overcome some of the inherent deficiencies of the traditional SDLC".[5]

Systems development phases

The System Development Life Cycle framework provides a sequence of activities for system designers and developers to follow. It consists of a set of steps or phases in which each phase of the SDLC uses the results of the previous one.

A Systems Development Life Cycle (SDLC) adheres to important phases that are essential for developers, such as planning, analysis, design, and implementation, and are explained in the section below. A number of system development life cycle (SDLC) models have been created: waterfall, fountain, spiral, build and fix, rapid prototyping, incremental, and synchronize and stabilize. The oldest of these, and the best known, is the waterfall model: a sequence of stages in which the output of each stage becomes the input for the next. These stages can be characterized and divided up in different ways, including the following[6]:

- **Preliminary Analysis**: The objective of phase1 is to conduct a preliminary analysis, propose alternative solutions, describe costs and benefits and submit a preliminary plan with recommendations.

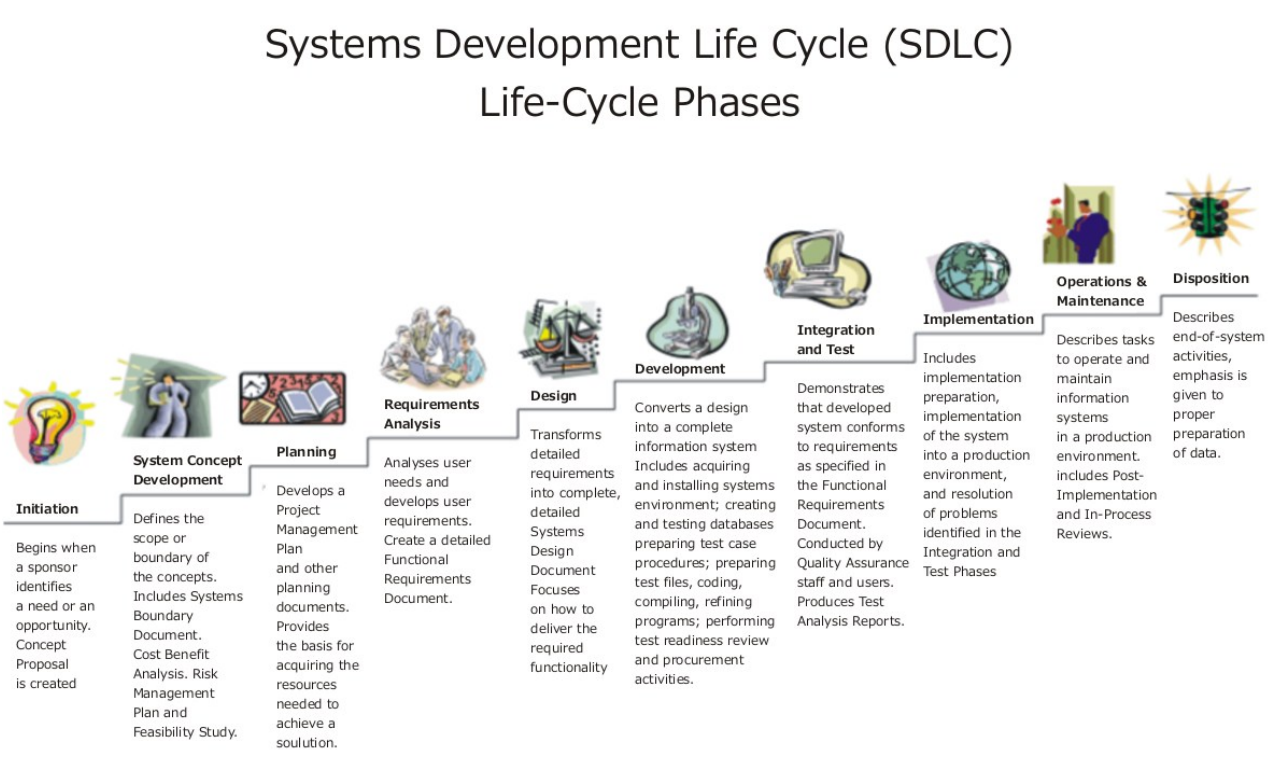
  Conduct the preliminary analysis: in this step, you need to find out the organization's objectives and the nature and scope of the problem under study. Even if a problem refers only to a small segment of the organization itself then you need to find out what the objectives of the organization itself are. Then you need to see how the problem being studied fits in with them.

  Propose alternative solutions: In digging into the organization's objectives and specific problems, you may have already covered some solutions. Alternate proposals may come from interviewing employees, clients , suppliers, and/or consultants. You can also study what competitors are doing. With this data, you will have three choices: leave the system as is, improve it, or develop a new system.

  Describe the costs and benefits.

- **Systems analysis, requirements definition**: Defines project goals into defined functions and operation of the intended application. Analyzes end-user information needs.
- **Systems design**: Describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudocode and other documentation.
- **Development**: The real code is written here.
- **Integration and testing**: Brings all the pieces together into a special testing environment, then checks for errors, bugs and interoperability.
- **Acceptance, installation, deployment**: The final stage of initial development, where the software is put into production and runs actual business.
- **Maintenance**: What happens during the rest of the software's life: changes, correction, additions, moves to a different computing platform and more. This is often the longest of the stages.

In the following example (see picture) these stage of the systems development life cycle are divided in ten steps from definition to creation and modification of IT work products:



The tenth phase occurs when the system is disposed of and the task performed is either eliminated or transferred to other systems. The tasks and work products for each phase are described in subsequent chapters.[7]

Not every project will require that the phases be sequentially executed. However, the phases are interdependent. Depending upon the size and complexity of the project, phases may be combined or may overlap.[7]

Investigation

The 1st stage of SDLC is the investigation phase. During this stage, business opportunities and problems are identified, and information technology solutions are discussed. Multiple alternative projects may be suggested and their feasibility analyzed. Operational feasibility is assessed, and it is determined whether or not the project fits with the current business environment, and to what degree it addresses business objects. In addition, an economic feasibility investigation is conducted to judge the costs and benefits of the project. Technical feasibility must also be

analyzed to determine if the available hardware and software resources are sufficient to meet expected specifications. A legal feasibility study is important to discover any potential legal ramification. The results of the feasibility study can then be compiled into a report, along with preliminary specifications. When the investigation stage ends, a decision whether or not to move forward with the project should be made. If it is decided to move ahead, a proposal should have been produced that outlines the general specifications of the project[8] .

System analysis

The goal of system analysis is to determine where the problem is in an attempt to fix the system. This step involves breaking down the system in different pieces to analyze the situation, analyzing project goals, breaking down what needs to be created and attempting to engage users so that definite requirements can be defined.

Design

In systems design the design functions and operations are described in detail, including screen layouts, business rules, process diagrams and other documentation. The output of this stage will describe the new system as a collection of modules or subsystems.

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts.

Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo-code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input design.

Testing

The code is tested at various levels in software testing. Unit, system and user acceptance testings are often performed. This is a grey area as many different opinions exist as to what the stages of testing are and how much, if any iteration occurs. Iteration is not generally part of the waterfall model, but usually some occur at this stage. In the testing the whole system is test one by one

Following are the types of testing:

- *Defect testing* the failed scenarios, including defect tracking
- Path testing
- Data set testing
- Unit testing
- System testing
- Integration testing
- Black-box testing
- White-box testing
- Regression testing
- Automation testing
- User acceptance testing
- Software performance testing

Operations and maintenance

The deployment of the system includes changes and enhancements before the decommissioning or sunset of the system. Maintaining the system is an important aspect of SDLC. As key personnel change positions in the organization, new changes will be implemented, which will require system.

Systems analysis and design

The **Systems Analysis and Design (SAD)** is the process of developing Information Systems (IS) that effectively use hardware, software, data, processes, and people to support the company's business objectives.

Object-oriented analysis

Object-oriented analysis (OOA) is the process of analyzing a task (also known as a problem domain), to develop a conceptual model that can then be used to complete the task. A typical OOA model would describe computer software that could be used to satisfy a set of customer-defined requirements. During the analysis phase of problem-solving, a programmer might consider a written requirements statement, a formal vision document, or interviews with stakeholders or other interested parties. The task to be addressed might be divided into several subtasks (or domains), each representing a different business, technological, or other areas of interest. Each subtask would be analyzed separately. Implementation constraints, (e.g., concurrency, distribution, persistence, or how the system is to be built) are not considered during the analysis phase; rather, they are addressed during object-oriented design (OOD).

The conceptual model that results from OOA will typically consist of a set of use cases, one or more UML class diagrams, and a number of interaction diagrams. It may also include some kind of user interface mock-up.

Input (sources) for object-oriented design

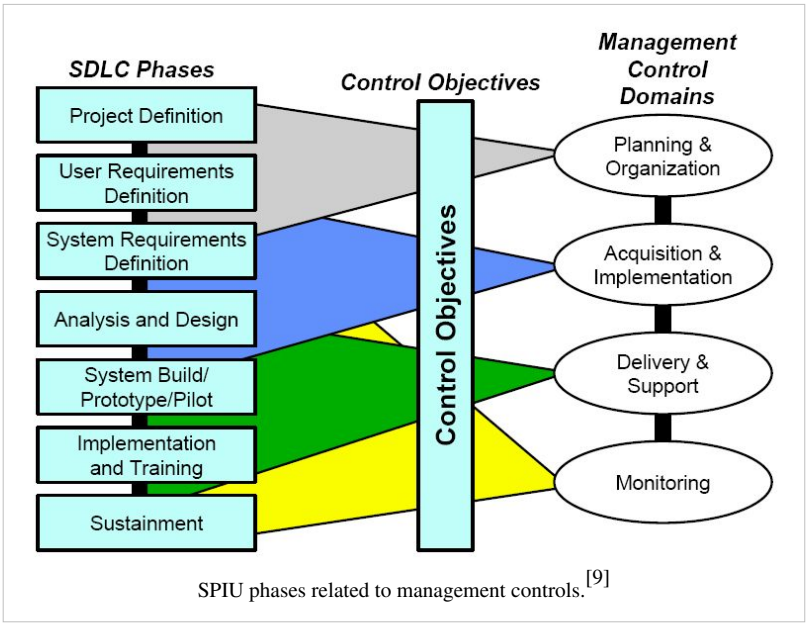
The input for object-oriented design is provided by the output of object-oriented analysis. Realize that an output artifact does not need to be completely developed to serve as input of object-oriented design; analysis and design may occur in parallel, and in practice the results of one activity can feed the other in a short feedback cycle through an iterative process. Both analysis and design can be performed incrementally, and the artifacts can be continuously grown instead of completely developed in one shot. sZDfas Some typical input artifacts for object-oriented design are:

- Conceptual model: Conceptual model is the result of object-oriented analysis, it captures concepts in the problem domain. The conceptual model is explicitly chosen to be independent of implementation details, such as concurrency or data storage.
- Use case: Use case is a description of sequences of events that, taken together, lead to a system doing something useful. Each use case provides one or more scenarios that convey how the system should interact with the users called actors to achieve a specific business goal or function. Use case actors may be end users or other systems. In many circumstances use cases are further elaborated into use case diagrams. Use case diagrams are used to identify the actor (users or other systems) and the processes they perform.
- System Sequence Diagram: System Sequence diagram (SSD) is a picture that shows, for a particular scenario of a use case, the events that external actors generate, their order, and possible inter-system events.
- User interface documentations (if applicable): Document that shows and describes the look and feel of the end product's user interface. It is not mandatory to have this, but it helps to visualize the end-product and therefore helps the designer.
- Relational data model (if applicable): A data model is an abstract model that describes how data is represented and used. If an object database is not used, the relational data model should usually be created before the design,

since the strategy chosen for object-relational mapping is an output of the OO design process. However, it is possible to develop the relational data model and the object-oriented design artifacts in parallel, and the growth of an artifact can stimulate the refinement of other artifacts.

Systems development life cycle

Management and control



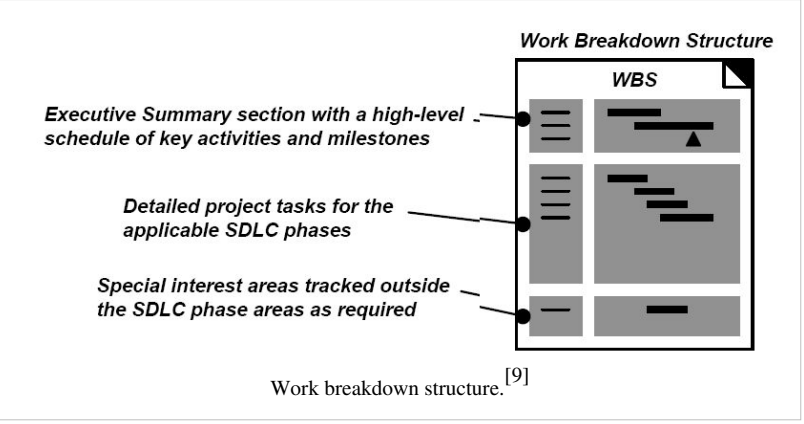
SPIU phases related to management controls.[9]

The SDLC phases serve as a programmatic guide to project activity and provide a flexible but consistent way to conduct projects to a depth matching the scope of the project. Each of the SDLC phase objectives are described in this section with key deliverables, a description of recommended tasks, and a summary of related control objectives for effective management. It is critical for the project manager to establish and monitor control objectives during each SDLC phase while executing projects. Control objectives help to provide a clear statement of the desired result or purpose and should be used throughout the entire SDLC process. Control objectives can be grouped into major categories (domains), and relate to the SDLC phases as shown in the figure.[9]

To manage and control any SDLC initiative, each project will be required to establish some degree of a Work Breakdown Structure (WBS) to capture and schedule the work necessary to complete the project. The WBS and all programmatic material should be kept in the “project description” section of the project notebook. The WBS format is mostly left to the project manager to establish in a way that best describes the project work there are many steps to find the java and c++ ways of drama languages.

```
There are some key areas that must be defined in the WBS as part of the
 SDLC policy. The following diagram describes three key areas that will
 be addressed in the WBS in a manner established by the project
manager.[9]
```

Work breakdown structured organization



Work breakdown structure.[9]

The upper section of the work breakdown structure (WBS) should identify the major phases and milestones of the project in a summary fashion. In addition, the upper section should provide an overview of the full scope and timeline of the project and will be part of the initial project description effort leading to project approval. The middle section of the WBS is based on the seven systems development life cycle (SDLC) phases as a guide for WBS task development. The WBS elements should consist of milestones and "tasks" as opposed to "activities" and have a definitive period (usually two weeks or more). Each task must have a measurable output (e.x. document, decision, or analysis). A WBS task may rely on one or more activities (e.g. software engineering, systems engineering) and may require close coordination with other tasks, either internal or external to the project. Any part of the project needing support from contractors should have a statement of work (SOW) written to include the appropriate tasks from the SDLC phases. The development of a SOW does not occur during a specific phase of SDLC but is developed to include the work from the SDLC process that may be conducted by external resources such as contractors and struct.[9]

Baselines in the SDLC

Baselines are an important part of the systems development life cycle (SDLC). These baselines are established after four of the five phases of the SDLC and are critical to the iterative nature of the model .[10] Each baseline is considered as a milestone in the SDLC.

- functional baseline: established after the conceptual design phase.
- allocated baseline: established after the preliminary design phase.
- product baseline: established after the detail design and development phase.
- updated product baseline: established after the production construction phase.

Complementary to SDLC

Complementary software development methods to systems development life cycle (SDLC) are:

- Software prototyping
- Joint applications development (JAD)
- Rapid application development (RAD)
- Extreme programming (XP); extension of earlier work in Prototyping and RAD.
- Open-source development
- End-user development
- Object-oriented programming

Comparison of Methodology Approaches (Post, & Anderson 2006)[11]

	SDLC	RAD	Open source	Objects	JAD	Prototyping	End User
Control	Formal	MIS	Weak	Standards	Joint	User	User
Time frame	Long	Short	Medium	Any	Medium	Short	Short –
Users	Many	Few	Few	Varies	Few	One or two	One
MIS staff	Many	Few	Hundreds	Split	Few	One or two	None
Transaction/DSS	Transaction	Both	Both	Both	DSS	DSS	DSS
Interface	Minimal	Minimal	Weak	Windows	Crucial	Crucial	Crucial
Documentation and training	Vital	Limited	Internal	In Objects	Limited	Weak	None
Integrity and security	Vital	Vital	Unknown	In Objects	Limited	Weak	Weak
Reusability	Limited	Some	Maybe	Vital	Limited	Weak	None

Strengths and weaknesses

Few people in the modern computing world would use a strict waterfall model for their systems development life cycle (SDLC) as many modern methodologies have superseded this thinking. Some will argue that the SDLC no longer applies to models like Agile computing, but it is still a term widely in use in technology circles. The SDLC practice has advantages in traditional models of software development, that lends itself more to a structured environment. The disadvantages to using the SDLC methodology is when there is need for iterative development or (i.e. web development or e-commerce) where stakeholders need to review on a regular basis the software being designed. Instead of viewing SDLC from a strength or weakness perspective, it is far more important to take the best practices from the SDLC model and apply it to whatever may be most appropriate for the software being designed.

A comparison of the strengths and weaknesses of SDLC:

Strength and Weaknesses of SDLC

Strengths	Weaknesses
Control.	Increased development time.
Monitor large projects.	Increased development cost.
Detailed steps.	Systems must be defined up front.
Evaluate costs and completion targets.	Rigidity.
Documentation.	Hard to estimate costs, project overruns.
Well defined user input.	User input is sometimes limited.
Ease of maintenance.	
Development and design standards.	
Tolerates changes in MIS staffing.	

An alternative to the SDLC is rapid application development, which combines prototyping, joint application development and implementation of CASE tools. The advantages of RAD are speed, reduced development cost, and active user involvement in the development process.

References

[1] SELECTING A DEVELOPMENT APPROACH (http://www.cms.hhs.gov/SystemLifecycleFramework/Downloads/SelectingDevelopmentApproach.pdf). Retrieved 27 October 2008.

[2] "Systems Development Life Cycle" (http://foldoc.org/foldoc.cgi?Systems+Development+Life+Cycle). In: Foldoc(2000-12-24)

[3] Software Development Life Cycle (SDLC) (http://docs.google.com/viewer?a=v&q=cache:bfhOl8jp1S8J:condor.depaul.edu/~jpetlick/extra/394/Session2.ppt+&hl=en&pid=bl&srcid=ADGEEShCfW0_MLC4wRbczfUxrndHTkbwguF9fZuaUCe0RDyOCWyO2PTmaPhHnZ4jRhZZ75maVO_7gVAD2ex5-QIhrj1683hMefBNkak7FkQJCAv sig=AHIEtbRhMlZ-TUyioKEhLQQxXk1WoSJXWA), Power Point, – Powered by Google Docs

[4] James Taylor (2004). *Managing Information Technology Projects*. p.39..

[5] Geoffrey Elliott & Josh Strachan (2004) *Global Business Information Technology*. p.87.

[6] QuickStudy: System Development Life Cycle (http://www.computerworld.com/s/article/71151/System_Development_Life_Cycle), By Russell Kay, May 14, 2002

[7] US Department of Justice (2003). INFORMATION RESOURCES MANAGEMENT (http://www.usdoj.gov/jmd/irm/lifecycle/ch1.htm) Chapter 1. Introduction.

[8] Marakas & O'Brien (2011). *Management Information Systems*. New York, NY: McGraw-Hill/Irwin. pp. 485-489. ISBN 978-0-07-337681-3.

[9] U.S. House of Representatives (1999). *Systems Development Life-Cycle Policy* (http://www.house.gov/cao-opp/PDFSolicitations/SDLCPOL.pdf). p.13.

[10] Blanchard, B. S., & Fabrycky, W. J.(2006) *Systems engineering and analysis* (4th ed.) New Jersey: Prentice Hall. p.31

[11] Post, G., & Anderson, D., (2006). *Management information systems: Solving business problems with information technology*. (4th ed.). New York: McGraw-Hill Irwin.

Further reading

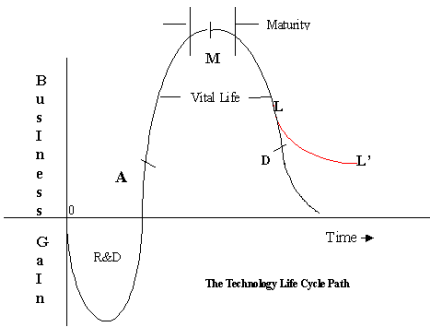
- Blanchard, B. S., & Fabrycky, W. J.(2006) *Systems engineering and analysis* (4th ed.) New Jersey: Prentice Hall.
- Cummings, Haag (2006). *Management Information Systems for the Information Age*. Toronto, McGraw-Hill Ryerson
- Beynon-Davies P. (2009). *Business Information Systems*. Palgrave, Basingstoke. ISBN 978-0-230-20368-6
- Computer World, 2002 (http://www.computerworld.com/developmenttopics/development/story/0,10801,71151,00.html), Retrieved on June 22, 2006 from the World Wide Web:
- Management Information Systems, 2005 (http://www.cbe.wwu.edu/misclasses/MIS320_Spring06_Bajwa/Chap006.ppt), Retrieved on June 22, 2006 from the World Wide Web:
- This article is based on material taken from the Free On-line Dictionary of Computing prior to 1 November 2008 and incorporated under the "relicensing" terms of the GFDL, version 1.3 or later.

External links

- The Agile System Development Lifecycle (http://www.ambysoft.com/essays/agileLifecycle.html)
- Pension Benefit Guaranty Corporation – Information Technology Solutions Lifecycle Methodology (http://www.pbgc.gov/docs/ITSLCM V2007.1.pdf)
- FSA Life Cycle Framework (http://federalstudentaid.ed.gov/static/gw/docs/lcm/FSALCMFrameworkOverview.pdf)
- HHS Enterprise Performance Life Cycle Framework (http://www.hhs.gov/ocio/eplc/eplc_framework_v1point2.pdf)
- The Open Systems Development Life Cycle (http://OpenSDLC.org)

Technology Life Cycle

The **technology life-cycle** (TLC) describes the commercial gain of a product through the expense of research and development phase, and the financial return during its "vital life". Some technologies, such as steel, paper or cement manufacturing, have a long lifespan (with minor variations in technology incorporated with time) whilst in other cases, such as electronic or pharmaceutical products, the lifespan may be quite short.



The typical life-cycle of a manufacturing process or production system from the stages of its initial conception to its culmination as either a technique or procedure of common practice or to its demise. The Y-axis of the diagram shows the business gain to the proprietor of the technology while the X-axis traces its lifetime.

The TLC associated with a product or technological service is different from product life-cycle (PLC) dealt with in product life-cycle management. The latter is concerned with the life of a product in the market-place in respect of timing of introduction, marketing measures and business costs. The *technology* underlying the product (such as, for example, that of a uniquely flavored tea) may be quite marginal but the process of creating and managing its life as a branded product will be very different.

The technology life cycle is concerned with the time and cost of developing the technology, the timeline of recovering cost and modes of making the technology yield a profit proportionate to the costs and risks involved. The TLC may, further, be protected during its cycle with patents and trademark seeking to lengthen the cycle and to maximize the profit from it.

The 'product' of the technology may just be a commodity such as the polyethylene plastic or a sophisticated product like the ICs used in a smartphone.

The development of a *competitive product* or process can have a major effect on the lifespan of the technology, making it shorter. Equally, the loss of patent rights through litigation, or loss of its secret elements (if any) through leakages also work to reduce its lifespan. Thus, it is apparent that the 'management' of the TLC is an important aspect of technology development.

In the simplest formulation, innovation can be thought of as being composed of research, development, demonstration, and deployment.[1]

The four phases of the technology life-cycle

The TLC may be seen as composed of four phases:

(a) the **research and development (R&D)** phase (sometimes called the "bleeding edge") when incomes from inputs are negative and where the prospects of failure are high

(b) the **ascent** phase when out-of-pocket costs have been recovered and the technology begins to gather strength by going beyond some Point A on the TLC (sometimes called the "leading edge")

(c) the **maturity** phase when gain is high and stable, the region, going into saturation, marked by M, and

(d) the **decline** (or decay phase), after a Point D, of reducing fortunes and utility of the technology.

Licensing options

In current world trends, with TLCs shortening due to competition and rapid innovation, a technology becomes technically licensable at all points of the TLC, whereas earlier, it was licensed only when it was past its maturity stage.

Large corporations develop technology for their own benefit and not with the objective of licensing. The tendency to license out technology only appears when there is a threat to the life of the TLC (business gain) as discussed later.

Licensing in the R&D phase

There are always smaller firms (SMEs) who are inadequately situated to finance the development of innovative R&D in the post-research and early technology phases. By sharing incipient technology under certain conditions, substantial risk financing can come from third parties. This is a form of quasi-licensing which takes different formats.

Even large corporates may not wish to bear all costs of development in areas of significant and high risk (e.g. aircraft development) and may seek means of spreading it to the stage that proof-of-concept is obtained.

In the case of small and medium firms, entities such as venture capitalists ('angels'), can enter the scene and help to materialize technologies. Venture capitalists accept both the costs and uncertainties of R&D, and that of market acceptance, in reward for high returns when the technology proves itself. Apart from finance, they may provide networking, management and marketing support. Venture capital connotes financial as well as human capital.

Large firms may opt for Joint R&D or work in a consortium for the early phase of development. Such vehicles are called strategic alliances – strategic partnerships.

With both venture capital funding and strategic (research) alliances, when business gains begin to neutralize development costs (the TLC crosses the X-axis), the ownership of the technology starts to undergo change.

In the case of smaller firms, venture capitalists help clients enter the stockmarket for obtaining substantially larger funds for development, maturation of technology, product promotion and to meet marketing costs. A major route is through initial public offering (IPO) which invite risk funding by the public for potential high gain. At the same time, the IPOs enable venture capitalists to attempt to recover expenditures already incurred by them through part sale of the stock pre-alloted to them (subsequent to the listing of the stock on the stock exchange). When the IPO is fully subscribed, the assisted enterprise becomes a corporation and can more easily obtain bank loans, etc. if needed.

Strategic alliance partners, allied on research, pursue separate paths of development with the incipient technology of common origin but pool their accomplishments through instruments such as 'cross-licensing'. Generally, contractual provisions among the members of the consortium allow a member to exercise the option of independent pursuit after joint consultation; in which case the optee owns all subsequent development.

Licensing in the ascent phase

The ascent stage of the technology usually refers to some point above Point A in the TLC diagram but actually it commences when the R&D portion of the TLC curve inflects (only that the cashflow is negative and unremunerative to Point A). The ascent is the strongest phase of the TLC because it is here that the technology is superior to alternatives and can command premium profit or gain. The slope and duration of the ascent depends on competing technologies entering the domain, although they may not be *as successful* in that period. Strongly patented technology extends the duration period.

The TLC begins to flatten out (the region shown as M) when equivalent or challenging technologies come into the competitive space and begin to eat away marketshare.

Till this stage is reached, the technology-owning firm would tend to exclusively enjoy its profitability, preferring *not* to license it. If an overseas opportunity does present itself, the firm would prefer to set up a controlled subsidiary

rather than license a third party.

Licensing in the maturity phase

The maturity phase of the technology is a period of stable and remunerative income but its competitive viability can persist over the larger timeframe marked by its 'vital life'. However, there may be a tendency to license out the technology to a third-parties during this stage to lower risk of decline in profitability (or competitivity) and to expand financial opportunity.

The exercise of this option is, generally, inferior to seeking participatory exploitation; in other words, engagement in joint venture, typically in regions where the technology would be in the *ascent phase*,as say, a developing country. In addition to providing financial opportunity it allows the technology-owner a degree of control over its use. Gain flows from the two streams of investment-based and royalty incomes. Further, the vital life of the technology is enhanced in such strategy.

Licensing in the decline phase

After reaching a point such as D in the above diagram, the earnings from the technology begin to decline rather rapidly. To prolong the life cycle, owners of technology might try to license it out at some point L when it can still be attractive to firms in other markets. This, then, traces the lengthening path, LL'. Further, since the decline is the result of competing rising technologies in this space, licenses may be attracted to the general lower cost of the older technology (than what prevailed during its vital life).

Licenses obtained in this phase are 'straight licenses'. They are free of direct control from the owner of the technology (as would otherwise apply,say, in the case of a joint-venture). Further, there may be fewer restrictions placed on the licensee in the employment of the technology.

The utility, viability, and thus the cost of straight-licenses depends on the estimated 'balance life' of the technology. For instance, should the key patent on the technology have expired, or would expire in a short while, the residual viability of the technology may be limited, although balance life may be governed by other criteria viz. knowhow which could have a longer life if properly protected.

It is important to note that the license has no way of knowing the stage at which the prime, and competing technologies, are on their TLCs. It would, of course, be evident to competing licensor firms, and to the originator, from the growth, saturation or decline of the profitability of their operations.

The license may, however, be able to approximate the stage by vigorously negotiating with the licensor and competitors to determine costs and licensing terms. A lower cost, or easier terms, *may* imply a declining technology.

In any case, access to technology in the decline phase is a large risk that the licensee accepts. (In a joint-venture this risk is substantially reduced by licensor sharing it). Sometimes, financial guarantees from the licensor may work to reduce such risk and can be negotiated.

There are instances when, even though the technology declines to becoming a technique, it may still contain important knowledge or experience which the licensee firm cannot learn of without help from the originator. This is often the form that *technical service* and *technical assistance* contracts take (encountered often in developing country contracts). Alternatively, consulting agencies may fill this role.

Technology development cycle

Technology development cycle describes the process of a new technology through the stages of technological maturity:

- Research and development
- demonstration
- deployment
- diffusion
- commercial maturity

References

[1] http://www.eoearth.org/article/Technological_innovation

- Manual on Technology Transfer Negotiation, United Nations Industrial Development Organization (A Reference for Policy-Makers and Practitioners on Technology Transfer. UNIDO, Vienna 1996, UNIDO Publication, ISBN 92-1-106302-7

Whole-life cost

Whole-life cost, or **Life-cycle cost** (LCC), refers to the total cost of ownership over the life of an asset.[1] Also commonly referred to as "cradle to grave" or "womb to tomb" costs. Costs considered include the financial cost which is relatively simple to calculate and also the environmental and social costs which are more difficult to quantify and assign numerical values. Typical areas of expenditure which are included in calculating the whole-life cost include, planning, design, construction and acquisition, operations, maintenance, renewal and rehabilitation, depreciation and cost of finance and replacement or disposal.

Financial

Whole-life cost analysis is often used for option evaluation when procuring new assets and for decision-making to minimise whole-life costs throughout the life of an asset.It is also applied to comparisons of actual costs for similar asset types and as feedback into future design and acquisition decisions.

The primary benefit is that costs which occur after an asset has been constructed or acquired, such as maintenance, operation, disposal, become an important consideration in decision-making. Previously, the focus has been on the up-front capital costs of creation or acquisition, and organisations may have failed to take account of the longer-term costs of an asset. It also allows an analysis of business function interrelationships. Low development costs may lead to high maintenance or customer service costs in the future.

Environmental and social

The use of environmental costs in a whole-life analysis allows a true comparison options, especially where both are quoted as "good" for the environment. For a major project such as the construction of a nuclear power station it is possible to calculate the environmental impact of making the concrete containment, the water required for refining the copper for the power plants and all the other components. Only by undertaking such an analysis is it possible to determine whether one solution carries a lower or higher environmental cost than another.[2]

Almost all major projects have some social impact. This may be the compulsory re-location of people living on land about to be submerged under a reservoir or a threat to the livelihood of small traders from the development of a hypermarket nearby.

Whole-life cost topics

Project appraisal

Whole-life costing is a key component in the economic appraisal associated with evaluating asset acquisition proposals. An economic appraisal is generally a broader based assessment, considering benefits and indirect or intangible costs as well as direct costs.

In this way, the whole-life costs and benefits of each option are considered and usually converted using discount rates into net present value costs and benefits. This results in a benefit cost ratio for each option, usually compared to the "do-nothing" counterfactual. Typically the highest benefit-cost ratio option is chosen as the preferred option.

Historically, asset investments have been based on expedient design and lowest cost construction. If such investment has been made without proper analysis of the standard of service required and the maintenance and intervention options available, the initial saving may result in increased expenditure throughout the asset's life.

By using whole-life costs, this avoids issues with decisions being made based on the short-term costs of design and construction. Often the longer-term maintenance and operation costs can be a significant proportion of the whole-life cost.

Asset management

During the life of the asset, decisions about how to maintain and operate the asset need to be taken in context with the effect these activities might have on the residual life of the asset. If by investing 10% more per annum in maintenance costs the asset life can be doubled, this might be a worthwhile investment.

Other issues which influence the lifecycle costs of an asset include:

- site conditions,
- historic performance of assets or materials,
- effective monitoring techniques,
- appropriate intervention strategies.

Although the general approach to determining whole-life costs is common to most types of asset, each asset will have specific issues to be considered and the detail of the assessment needs to be tailored to the importance and value of the asset. High cost assets (and asset systems) will likely have more detail, as will critical assets and asset systems.

Maintenance expenditure can account for many times the initial cost of the asset. Although an asset may be constructed with a design life of 30 years, in reality it will possibly perform well beyond this design life. For assets like these a balanced view between maintenance strategies and renewal/rehabilitation is required. The appropriateness of the maintenance strategy must be questioned, the point of intervention for renewal must be challenged. The process requires proactive assessment which must be based on the performance expected of the asset, the consequences and probabilities of failures occurring, and the level of expenditure in maintenance to keep the service available and to avert disaster.

IT industry usage

Whole-life cost is often referred to as "total cost of ownership (TCO)" when applied to IT hardware and software acquisitions. Use of the term "TCO" appears to have been popularised by Gartner Group in 1987[3] but its roots are considerably older, dating at least to the first quarter of the twentieth century.[4]

It has since been developed as a concept with a number of different methodologies and software tools. A TCO assessment ideally offers a final statement reflecting not only the cost of purchase but all aspects in the further use and maintenance of the equipment, device, or system considered. This includes the costs of training support personnel and the users of the system, costs associated with failure or outage (planned and unplanned), diminished performance incidents (i.e. if users are kept waiting), costs of security breaches (in loss of reputation and recovery costs), costs of disaster preparedness and recovery, floor space, electricity, development expenses, testing infrastructure and expenses, quality assurance, boot image control, marginal incremental growth, decommissioning, e-waste handling, and more. When incorporated in any financial benefit analysis (e.g., ROI, IRR, EVA, ROIT, RJE) TCO provides a cost basis for determining the economic value of that investment.

Understanding and familiarity with the term TCO has been somewhat facilitated as a result of various comparisons between the TCO of open source and proprietary software. Because the software cost of open source software is often zero, TCO has been used as a means to justify the up-front licensing costs of proprietary software. Studies which attempt to establish the TCO and provide comparisons have as a result been the subject of many discussions regarding the accuracy or perceived bias in the comparison.

Automobile industry, finances

Total cost of ownership is also common in the automobile industry. In this context, the TCO denotes the cost of owning a vehicle from the purchase, through its maintenance, and finally its sale as a used car. Comparative TCO studies between various models help consumers choose a car to fit their needs and budget.

TCO can and often does vary dramatically against TCA (total cost of acquisition), although TCO is far more relevant in determining the viability of any capital investment, especially with modern credit markets and financing. TCO also directly relates to a business's total costs across all projects and processes and, thus, its profitability. Some instances of "TCO" appear to refer to "total cost of operation", but this may be a subset of the total cost of ownership if it excludes maintenance and support costs.

References

[1] Association of Local Government Engineers New Zealand: "Infrastructure Asset Management Manual", June 1998 - Edition 1.1

[2] Whole Life Costing For Sustainable Drainage (http://www.ciria.org.uk/suds/pdf/whole_life_cost_summary.pdf)

[3] About Gartner TCO (http://amt.gartner.com/TCO/MoreAboutTCO.htm)

[4] TCO: What's Old is New (http://www.processor.com/editorial/article.asp?article=articles/p3012/06p12/06p12.asp&guid=38EA5B42565B4C989AB66754B695F44C&searchtype=&WordList=&bJumpTo=True)

Further reading

- Riggs, James L., (1982), *Engineering economics*. McGraw-Hill, New York, 2nd edition, 1982.
- Norris, G. A. (2001): Integrating Life Cycle Cost Analysis and LCA, in: The International Journal of Life Cycle Assessment, Jg. 6, H. 2, p. 118–120.
- Schaltegger, S. & Burritt, R. (2000): Contemporary Environmental Accounting. Issues, Concepts and Practice. Sheffield: Greenleaf Publ.
- Kicherer, A.; Schaltegger, S.; Tschochohei, H. & Ferreira Pozo, B.: Eco-Efficiency. Combining Life Cycle Assessment and Life Cycle Costs via Normalization, International Journal of LCA, 2007, Vol 12, No 7, 537-543.

External links

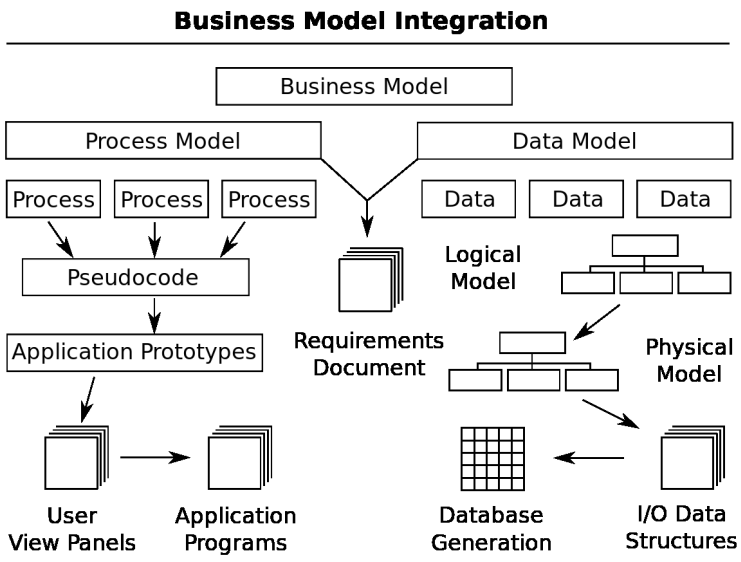
- Whole-life cost forum (http://www.wlcf.org.uk)
- Whole-life costing for sustainable drainage (http://www.ciria.org.uk/suds/pdf/whole_life_cost_summary.pdf)
- [[BSRIA (http://www.BSRIA.co.uk/news/1886/)] article: "What is whole life cost analysis?"]
- Role of depreciation (http://www.cipfa.org.uk/panels/treasury_management/download/depreciation16july02.pdf)
- Cost Structure and Life Cycle Cost (LCC) for Military Systems - Papers presented at the RTO Studies, Analysis and Simulation Panel (SAS) Symposium held in Paris, France, 24-25 October 2001 (http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA418689#page=101)

Modelling

Enterprise modelling

Enterprise modelling is the abstract representation, description and definition of the structure, processes, information and resources of an identifiable business, government body, or other large organization.[2]

It deals with the process of understanding an enterprise business and improving its performance through creation of enterprise models. This includes the modelling of the relevant business domain (usually relatively stable), business processes (usually more volatile), and Information technology.



Graphical representation of some types of models in enterprise modelling. A business model illustrates the functions associated with a process that are performance and the organizations that perform these functions. In software development often both business process models and data models are being developed as part of the process of creating application programs on the one side and databases on the other side.[1]

Overview

Enterprise modelling is the process of building models of whole or part of an enterprise with process models, data models, resource models and or new ontologies etc. It is based on knowledge about the enterprise, previous models and/or reference models as well as domain ontologies using model representation languages.[3] An enterprise in general is a unit of economic organization or activity. These activities are required to develop and deliver products and/or services to a customer. An enterprise includes a number of functions and operations such as purchasing, manufacturing, marketing, finance, engineering, and research and development. The enterprise of interest are those corporate functions and operations necessary to manufacture current and potential future variants of a product.[4]

The term "enterprise model" is used in industry to represent differing enterprise representations, with no real standardized definition.[5] Due to the complexity of enterprise organizations, a vast number of differing enterprise modelling approaches have been pursued across industry and academia.[6] Enterprise modelling constructs can focus upon manufacturing operations and/or business operations; however, a common thread in enterprise modelling is an inclusion of assessment of information technology. For example, the use of networked computers to trigger and receive replacement orders along a material supply chain is an example of how information technology is used to coordinate manufacturing operations within an enterprise.[4]

The basic idea of enterprise modelling according to Ulrich Frank[7] is "to offer different views on an enterprise, thereby providing a medium to foster dialogues between various stakeholders - both in academia and in practice. For this purpose they include abstractions suitable for strategic planning, organisational (re-) design and software engineering. The views should complement each other and thereby foster a better understanding of complex systems by systematic abstractions. The views should be generic in the sense that they can be applied to any enterprise. At

the same time they should offer abstractions that help with designing information systems which are well integrated with a company's long term strategy and its organisation. Hence, enterprise models can be regarded as the conceptual infrastructure that support a high level of integration."[7]

History

Enterprise modelling has its roots in systems modelling and especially information systems modelling. One of the earliest pioneering works in modelling information systems has been done by Young and Kent (1958),[8][9] who argued for "a precise and abstract way of specifying the informational and time characteristics of a data processing problem". They wanted to create "a notation that should enable the analyst to organize the problem around any piece of hardware". Their work was a first efforts to create an abstract specification and invariant basis for designing different alternative implementations using different hardware components. A next step in IS modelling was taken by CODASYL, an IT industry consortium formed in 1959, who essentially aimed at the same thing as Young and Kent: the development of "a proper structure for machine independent problem definition language, at the system level of data processing". This led to the development of a specific IS information algebra.[9]

The first methods dealing with enterprise modelling emerged in the 1970s. They were the entity-relationship approach of Peter Chen (1976) and SADT of Douglas T. Ross (1977), the one concentrate on the information view and the other on the function view of business entities.[3] These first methods have been followed end 1970s by numerous methods for software engineering, such as SSADM, Structured Design, Structured Analysis and others. Specific methods for enterprise modelling in the context of Computer Integrated Manufacturing appeared in the early 1980s. They include the IDEF family of methods (ICAM, 1981) and the GRAI method by Doumeingts in 1984[10] followed by GRAI/GIM by Doumeingts and others in 1992.[11]

These second generation of methods were activity-based methods which have been surpassed on the one hand by process-centred modelling methods developed in the 1990s such as Architecture of Integrated Information Systems (ARIS), CIMOSA and Integrated Enterprise Modeling (IEM). And on the other hand by object-oriented methods, such as Object-oriented analysis (OOA) and Object-modelling technique (OMT).[3]

Enterprise modelling basics

Enterprise model

An enterprise model is a representation of the structure, activities, processes, information, resources, people, behavior, goals, and constraints of a business, government, or other enterprises.[12] Thomas Naylor (1970) defined a (simulation) model as "an attempt to describe the interrelationships among a corporation's financial, marketing, and production activities in terms of a set of mathematical and logical relationships which are programmed into the computer."[13] These interrelationships should according to Gershefski (1971) represent in detail all aspects of the firm including "the physical operations of the company, the accounting and financial practices followed, and the response to investment in key areas"[14] Programming the modelled relationships into the computer is not always necessary: enterprise models, under different names, have existed for centuries and were described, for example, by Adam Smith, Walter Bagehot, and many others.

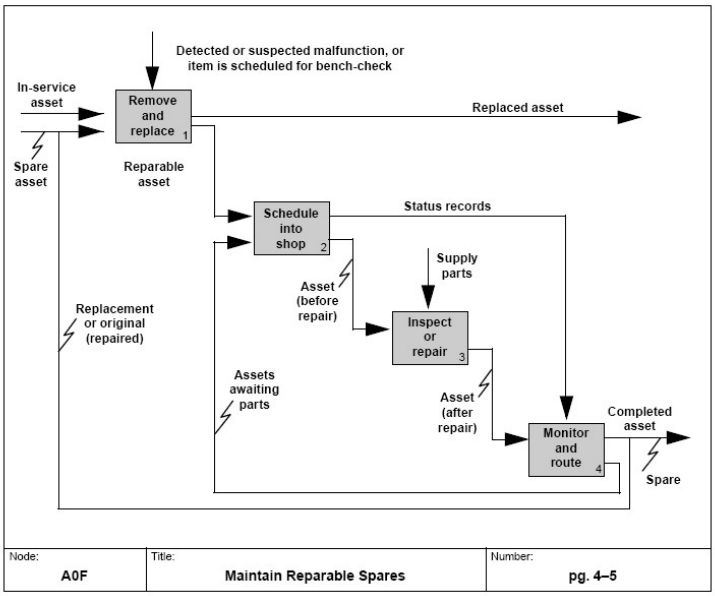
According to Fox and Gruninger (1998) from "a design perspective, an enterprise model should provide the language used to explicitly define an enterprise... From an operations perspective, the enterprise model must be able to represent what is planned, what might happen, and what has happened. It must supply the information and knowledge necessary to support the operations of the enterprise, whether they be performed by hand or machine."[12]

In a two-volume set entitled *The Managerial Cybernetics of Organization* Stafford Beer introduced a model of the enterprise, the Viable System Model (VSM). Volume 2, *The Heart of Enterprise,*[15] analyzed the VSM as a recursive organization of five systems: System One (S1) through System Five (S5). Beer's model differs from others in that the VSM is recursive, not hierarchical: "In a recursive organizational structure, any viable system contains,

and is contained in, a viable system."[15]

Function modelling

Function modelling in systems engineering is a structured representation of the functions, activities or processes within the modelled system or subject area.[16]



Example of a function model of the process of "Maintain Reparable Spares" in IDEF0 notation.

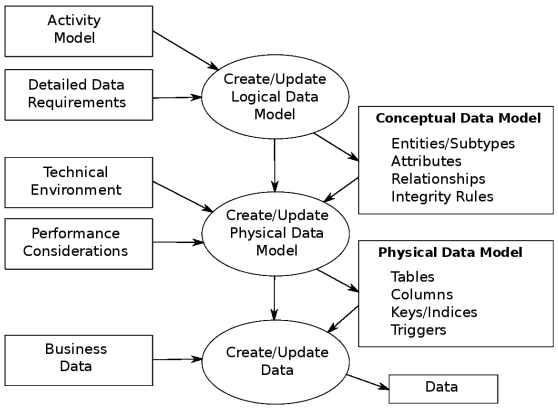
A function model, also called an activity model or process model, is a graphical representation of an enterprise's function within a defined scope. The purpose of the function model are to describe the functions and processes, assist with discovery of information needs, help identify opportunities, and establish a basis for determining product and service costs.[17] A function model is created with a functional modelling perspective. A functional perspectives is one or more perspectives possible in process modelling. Other perspectives possible are for example behavioural, organisational or informational.[18]

A functional modelling perspective concentrates on describing the dynamic process. The main concept in this modelling perspective is the process, this could be a function, transformation, activity, action, task etc. A well-known example of a modelling language employing this perspective is data flow diagrams. The perspective uses four symbols to describe a process, these being:

- Process: Illustrates transformation from input to output.
- Store: Data-collection or some sort of material.
- Flow: Movement of data or material in the process.
- External Entity: External to the modelled system, but interacts with it.

Now, with these symbols, a process can be represented as a network of these symbols. This decomposed process is a DFD, data flow diagram. In Dynamic Enterprise Modeling, for example, a division is made in the Control model, Function Model, Process model and Organizational model.

Data modelling

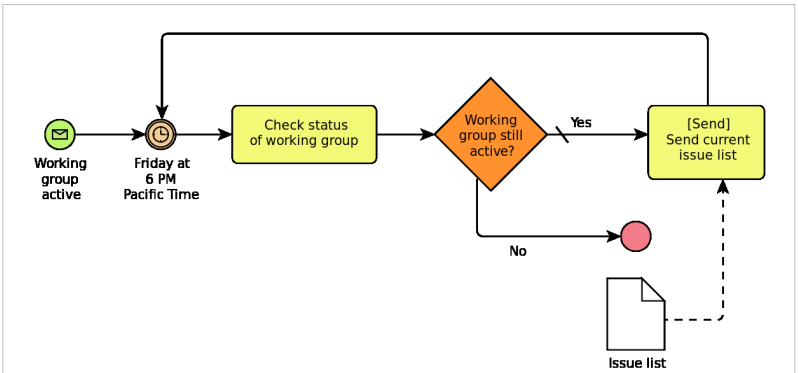


The data modelling process.

Data modelling is the process of creating a data model by applying formal data model descriptions using data modelling techniques. Data modelling is a technique for defining business requirements for a database. It is sometimes called *database modelling* because a data model is eventually implemented in a database.[19]

The figure illustrates the way data models are developed and used today. A conceptual data model is developed based on the data requirements for the application that is being developed, perhaps in the context of an activity model. The data model will normally consist of entity types, attributes, relationships, integrity rules, and the definitions of those objects. This is then used as the start point for interface or database design.[20]

Business process modelling



Example of business process modelling with the Business Process Modeling Notation.

Business process modelling (BPM) is the activity of representing processes of an enterprise, so that the current ("as is") process may be analyzed and improved in future ("to be"). Business process modelling is typically performed by business analysts and managers who are seeking to improve process efficiency and quality. The process improvements identified by business process modelling may or may not require Information Technology involvement, although that is a common driver for the need to model a business process, by creating a process master.

Change management programs are typically involved to put the improved business processes into practice. With advances in technology from large platform vendors, the vision of business process modelling models becoming fully executable (and capable of simulations and round-trip engineering) is coming closer to reality every day.

Systems architecture

The RM-ODP reference model identifies enterprise modelling as providing one of the five viewpoints of an open distributed system. Note that such a system need not be a modern-day IT system: a banking clearing house in the 19th century may be used as an example.

Enterprise modelling techniques

There are several techniques for modelling the enterprise such as

- Active Knowledge Modeling,[21]
- Design & Engineering Methodology for Organizations (DEMO)
- Dynamic Enterprise Modeling
- Enterprise Modelling Methodology/Open Distributed Processing (EMM/ODP)
- Extended Enterprise Modeling Language
- Multi-Perspective Enterprise Modelling (MEMO),[22]
- Process modelling such as CIMOSA, DYA, IDEF3, LOVEM, PERA, etc.
- Integrated Enterprise Modeling (IEM), and
- Modelling the enterprise with multi-agent systems.

More enterprise modelling techniques are developed into Enterprise Architecture framework such as:

- ARIS - ARchitecture of Integrated Information Systems
- DoDAF - the US Department of Defense Architecture Framework
- OBASHI The OBASHI Business & IT methodology and framework
- RM-ODP - Reference Model of Open Distributed Processing
- TOGAF - The Open Group Architecture Framework
- Zachman Framework - an architecture framework, based on the work of John Zachman at IBM in the 1980s

And metamodelling frameworks such as:

- Generalised Enterprise Reference Architecture and Methodology

Enterprise engineering

Enterprise engineering is the discipline concerning the design and the engineering of enterprises, regarding both their business and organization.[23] In theory and practice two types of enterprise engineering has emerged. A more general connected to engineering and the management of enterprises, and a more specific related to software engineering, enterprise modelling and enterprise architecture.

In the field of engineering a more general enterprise engineering emerged, defined[24] as the application of engineering principals to the management of enterprises. It encompasses the application of knowledge, principles, and disciplines related to the analysis, design, implementation and operation of all elements associated with an enterprise. In essence this is an interdisciplinary field which combines systems engineering and strategic management as it seeks to engineer the entire enterprise in terms of the products, processes and business operations. The view is one of continuous improvement and continued adaptation as firms, processes and markets develop along their life cycles. This total systems approach encompasses the traditional areas of research and development, product design, operations and manufacturing as well as information systems and strategic management.[24] This fields is related to engineering management, operations management, service management and systems engineering.

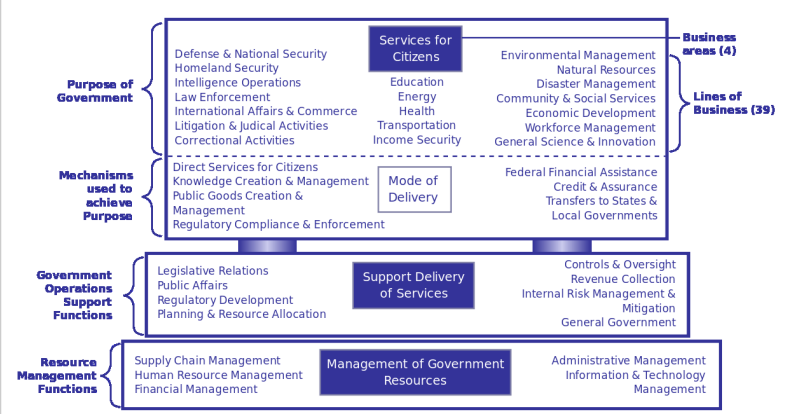
In the context of software development a specific field of enterprise engineering has emerged, which deals with the modelling and integration of various organizational and technical parts of business processes.[25] In the context of information systems development it has been the area of activity in the organization of the systems analysis, and an extension of the scope of Information Modelling.[26] It can also be viewed as the extension and generalization of the

systems analysis and systems design phases of the software development process.[27] Here Enterprise modelling can be part of the early, middle and late information system development life cycle. Explicit representation of the organizational and technical system infrastructure is being created in order to understand the orderly transformations of existing work practices.[27] This field is also called Enterprise architecture, or defined with Enterprise Ontology as being two major parts of Enterprise architecture.[23]

Related fields

Business reference modelling

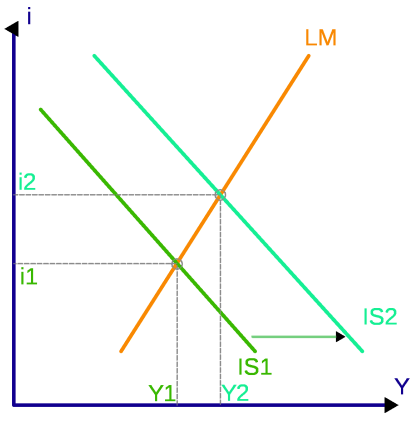
Business reference modelling is the development of reference models concentrating on the functional and organizational aspects of the core business of an enterprise, service organization or government agency. In enterprise engineering a business reference model is part of an enterprise architecture framework. This framework defines in a series of reference models, how to organize the structure and views associated with an Enterprise Architecture.



Example of the US FEA Business Reference Model.[28]

A reference model in general is a model of something that embodies the basic goal or idea of something and can then be looked at as a reference for various purposes. A business reference model is a means to describe the business operations of an organization, independent of the organizational structure that perform them. Other types of business reference model can also depict the relationship between the business processes, business functions, and the business area's business reference model. These reference model can be constructed in layers, and offer a foundation for the analysis of service components, technology, data, and performance.

Economic modelling



A diagram of the IS/LM model

Economic modelling is the theoretical representation of economic processes by a set of variables and a set of logical and/or quantitative relationships between them. The economic model is a simplified framework designed to illustrate complex processes, often but not always using mathematical techniques. Frequently, economic models use structural parameters. Structural parameters are underlying parameters in a model or class of models.[29] A model may have various parameters and those parameters may change to create various properties.[30]

In general terms, economic models have two functions: first as a simplification of and abstraction from observed data, and second as a means of selection of data based on a paradigm of econometric study. The simplification is particularly important for economics given the enormous complexity of economic processes. This complexity can be attributed to the diversity of factors that determine economic activity; these factors include: individual and cooperative decision processes, resource

limitations, environmental and geographical constraints, institutional and legal requirements and purely random fluctuations. Economists therefore must make a reasoned choice of which variables and which relationships between these variables are relevant and which ways of analyzing and presenting this information are useful.

Ontology engineering

Ontology engineering or *ontology building* is a subfield of knowledge engineering that studies the methods and methodologies for building ontologies. In the domain of enterprise architecture, an ontology is an outline or a schema used to structure objects, their attributes and relationships in a consistent manner.[4] As in enterprise modelling, an ontology can be composed of other ontologies. The purpose of ontologies in enterprise modelling is to formalize and establish the sharability, re-usability, assimilation and dissemination of information across all organizations and departments within an enterprise. Thus, an ontology enables integration of the various functions and processes which take place in an enterprise.[31]

One common language with well articulated structure and vocabulary would enable the company to be more efficient in its operations. A common ontology will allow for effective communication, understanding and thus coordination among the various divisions of an enterprise. There are various kinds of ontologies used in numerous environments. While the language example given earlier dealt with the area of information systems and design, other ontologies may be defined for processes, methods, activities, etc., within an enterprise.[4]

Using ontologies in enterprise modelling offers several advantages. Ontologies ensure clarity, consistency, and structure to a model. They promote efficient model definition and analysis. Generic enterprise ontologies allow for reusability of and automation of components. Because ontologies are schemata or outlines, the use of ontologies does not ensure proper enterprise model definition, analysis, or clarity. Ontologies are limited by how they are defined and implemented. An ontology may or may not include the potential or capability to capture all of the aspects of what is being modelled.[4]

Systems thinking

The modelling of the enterprise and its environment could facilitate the creation of enhanced understanding of the business domain and processes of the extended enterprise, and especially of the relations—both those that "hold the enterprise together" and those that extend across the boundaries of the enterprise. Since enterprise is a system, concepts used in system thinking[32] can be successfully reused in modelling enterprises.

This way a fast understanding can be achieved throughout the enterprise about how business functions are working and how they depend upon other functions in the organization.

References

[1] Paul R. Smith & Richard Sarfaty (1993). Creating a strategic plan for configuration management using Computer Aided Software Engineering (CASE) tools. (http://www.osti.gov/energycitations/servlets/purl/10160331-YhIRrY/) Paper For 1993 National DOE/Contractors and Facilities CAD/CAE User's Group.

[2] Cornelius T. Leondes, Richard Henry Frymuth Jackson (1992). *Manufacturing and Automation Systems: Techniques and Technologies*. Academic Press, 1992. ISBN 0-12-012745-8, p.97

[3] F.B. Vernadat (1997). Enterprise Modelling Languages ICEIMT'97 Enterprise Integration - International Consensus (http://www.mel.nist.gov/workshop/iceimt97/pap-ver3/pap-ver3.htm). EI-IC ESPRIT Project 21.859.

[4] James K. Ostie (1996). "An Introduction to Enterprise Modeling and Simulation" (http://www.osti.gov/bridge/servlets/purl/432930-yXrz5I/webviewable/432930.PDF)

[5] E. Aranow (1991). "Modeling Exercises Shape Up Enterprises". In: *Software Magazine* Vol.11 , p. 36-43

[6] C. J. Pétrie Jr. (1992). "Introduction", In: *Enterprise Integration Modeling - Proceedings of the First International Conference* MIT Press, p. 563.

[7] "Enterprise modeling" (http://www.wi-inf.uni-due.de/FGFrank/index.php?lang=en&&groupId=1&&contentType=ResearchInterest&&topicId=14) by Ulrich Frank (2009) at wi-inf.uni-due.de. Retrieved May 30, 2009.

[8] Young, J. W., and Kent, H. K. (1958). "Abstract Formulation of Data Processing Problems". In: *Journal of Industrial Engineering*. Nov-Dec 1958. 9(6), pp. 471-479

[9] Janis A. Bubenko jr (2007) "From Information Algebra to Enterprise Modelling and Ontologies - a Historical Perspective on Modelling for Information Systems". In: *Conceptual Modelling in Information Systems Engineering*. John Krogstie et al. eds. pp 1-18

[10] Doumeingts, G. (1984) *La Méthode GRAI*. PhD. Thesis, University of Bordeaux I, Bordeaux, France. (In French).

[11] Doumeingts, G., Vallespir, B., Zanettin, M. and Chen, D. (1992) *GIM, GRAI Integrated Methodology - A methodology for Designing CIM systems, Version 1.0*. Unnumbered Report, LAP/GRAI, University of Bordeaux I, France

[12] Mark S. Fox and Michael Gruninger (1998) " Enterprise Modeling (http://www.eil.utoronto.ca/enterprise-modelling/papers/fox-aimag98.pdf)". American Association for Artificial Intelligence.

[13] Naylor, T. (1970) Corporate simulation models and the economic theory of the firm, in Schrieber, A. (editor) "Corporate simulation models", University of Washington Press, Seattle, 1970, pp 1-35.

[14] Gershefski, G. (1971) "What's happening in the world of corporate models?", Interfaces, Vol 1, No 4. p.44

[15] Beer, Stafford. (1979) *The Heart of Enterprise,* Wiley.

[16] FIPS Publication 183 (http://www.itl.nist.gov/fipspubs/idef02.doc) released of IDEFØ December 1993 by the Computer Systems Laboratory of the National Institute of Standards and Technology (NIST).

[17] Reader's Guide to IDEF0 Function Models (http://www.archives.gov/era/pdf/rmsc-19951006-dod-rm-function-and-information-models.pdf). Accessed 27 Nov 2008.

[18] Process perspectives (http://users.jyu.fi/~jpt/ME2000/Me14/sld004.htm). In: *Metamodeling and method engineering*, Minna Koskinen, 2000.

[19] Whitten, Jeffrey L.; Lonnie D. Bentley, Kevin C. Dittman. (2004). *Systems Analysis and Design Methods*. 6th edition. ISBN 0-256-19906-X.

[20] Matthew West and Julian Fowler (1999). Developing High Quality Data Models (http://www.matthew-west.org.uk/documents/princ03.pdf). The European Process Industries STEP Technical Liaison Executive (EPISTLE).

[21] Frank Lillehagen, John Krogstie (2008). *Active Knowledge Modeling of Enterprises*. Springer, 2008. ISBN 3-540-79415-8

[22] Ulrich Frank (2002). " Multi-Perspective Enterprise Modeling (MEMO): Conceptual Framework and Modeling Languages (http://www.wi-inf.uni-essen.de/FGFrank/documents/Konferenzbeitraege/HICSS2002.pdf)". In: *Proceedings of the Hawaii International Conference on System Sciences (HICSS-35). Los Alamitos, CA*. Ralph H. Sprague, Jr. (eds.). IEEE Computer Society Press.

[23] Jan Dietz (2006). *Enterprise Ontology - Theory and Methodology*. Springer-Verlag Berlin Heidelberg.

[24] Enterprise Engineering Research at Royal Holloway (http://personal.rhul.ac.uk/uhtm/001/homepage.html) led by Dr Alan Pilkington, Ver 9.08. Accessed 4 November 2008.

[25] Vernadat, F.B. (1996) *Enterprise Modeling and Integration: Principles and Applications*. Chapman & Hall, London, ISBN 0-412-60550-3.

[26] J A Bubenko (1993). "Extending the Scope of Information Modelling". In: *Proceedings of the 4th International Workshop on the Deductive Approach to Information Systems and Databases, Costa Brava, Catalonia*. 1993.

[27] Gustas, R and Gustiene, P (2003) "Towards the Enterprise engineering approach for Information system modelling across organisational and technical boundaries", in: *Proceedings of the fifth International Conference on Enterprise Information Systems*, vol. 3, Angers, France, 2003, pp. 77-88.

[28] FEA (2005) FEA Records Management Profile, Version 1.0 (http://www.archives.gov/records-mgmt/pdf/rm-profile.pdf). December 15, 2005.

[29] Moffatt, Mike. (2008) About.com *Structural Parameters* (http://economics.about.com/od/economicsglossary/g/structuralp.htm) Economics Glossary; Terms Beginning with S. Accessed June 19, 2008.

[30] Moffatt, Mike. (2008) About.com *Structure* (http://economics.about.com/od/economicsglossary/g/structure.htm) Economics Glossary; Terms Beginning with S. Accessed June 19, 2008.

[31] G. Fadel, M. Fox, M. Gruninger (1994). "A Generic Enterprise Resource Ontology". In: *Proceedings of the 3rd Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*. p. 117-128

[32] (see, for example, (Weinberg, 1982), or, more generally, works by Bunge, for example, (Bunge, 2003) and by Hayek, for example, (Hayek, 1967))

Further reading

- August-Wilhelm Scheer (1992). *Architecture of Integrated Information Systems: Foundations of Enterprise Modelling*. Springer-Verlag. ISBN 3-540-55131-X
- François Vernadat (1996) *Enterprise Modeling and Integration: Principles and Applications*, Chapman & Hall, London, ISBN 0-412-60550-3

External links

- Agile Enterprise Modeling (http://www.agiledata.org/essays/enterpriseArchitecture.html). by S.W. Ambler, 2003-2008.
- Enterprise Modeling Anti-patterns (http://www.agilemodeling.com/essays/enterpriseModelingAntiPatterns.htm). by S.W. Ambler, 2005.

Collaboration

Business analyst

A **Business Analyst** (*BA*) is someone who analyzes the existing or ideal organization and design of systems, including businesses, departments, and organizations. BAs also assess business models and their integration with technology.

Levels

There are at least four tiers of business analysis:

1. Planning Strategically – The analysis of the organization's strategic business needs
2. Operating/Business Model Analysis – The definition and analysis of the organization's policies and market business approaches
3. Process Definition and Design – The business process modeling (often developed through process modeling and design)
4. IT/Technical Business Analysis – The interpretation of business rules and requirements for technical systems (generally within IT, sometimes referred to as Systems analysis)

Alternative descriptions

BCS, The Chartered Institute for IT, proposes the following definition of a business analyst: "An internal consultancy role that has responsibility for investigating business systems, identifying options for improving business systems and bridging the needs of the business with the use of IT." [1]

The International Institute of Business Analysis (IIBA) describes the role as: "a liaison among stakeholders in order to understand the structure, policies, and operations of an organization, and to recommend solutions that enable the organization to achieve its goals."[2]

The Certified Software Business Analyst (CSBA) Common Body of Knowledge, defines this as: "uniquely placed in the organization to provide a strong link between the Business Community and Information Technology (IT)."

The role of Business Analyst has evolved from someone who was a part of the business operation and worked with Information Technology to improve the quality of the products and services being delivered by the IT organization to someone who apart from gathering Business requirements, also assists in Integration and Acceptance Testing, supports the development of training and implementation material, participates in the implementation, and provides post-implementation support. Business Analysts today are also involved in the development of project plans and often provide project management skills when these skills are not available in other project participants.

Typical deliverables

Depending on the level of involvement of business analysis and the goal of the Project sponsor, the deliverable areas range from the Business requirements definition, Functional requirements definition (converting detailed business rules into system requirements), As-Is process definition, To-Be process definition to Business case (conversion of shareholder return and risk appetite into strategic plans).

The following section focuses on the IT sector perspective around business analysis, where much of the deliverables are around requirements. The BA will record requirements in some form of requirements management tool, whether a simple spreadsheet or a complex application.

Business requirements

(project initiation document), what the needed achievements will be, and the quality measures (i.e. KPIs or tangible objectives). They are usually expressed in terms of broad outcomes the business requires (i.e. Ability statements), rather than specific functions the system may perform. Specific design elements are usually outside the scope of this document, although design standards may be referenced.

- Example: The ability to accept and process customer feedback about the provided service.

Functional requirements

describe what the system, process, or product/service must do in order to fulfill the business requirements (i.e. the system shall ...). Note that the business requirements often can be broken up into sub-business requirements and many functional requirements. These are often referred to as System Requirements although some functionality could be manual and not system based, e.g., create notes or work instructions. A system does not necessary means an IT or computer program. A system could be a non-IT program (i.e. ICMS: a system based used within Fire fighting, where Fire fighters uses a plastic card system to indicate that they are combating at the incident and provides the Incident controller with a means of tracking his resources and safety).

- An example of business requirement:

1. The system shall provide the ability to associate notes to a project plan.
2. The system shall allow the user to enter free text to the project plan notes, up to 255 characters in length.

User (stakeholder) requirements

are a very important part of the deliverables, the needs of the stakeholders must be correctly documented. This deliverable can also reflect how the product will be designed and developed, and define how test cases must be formulated. However, stakeholders may not always be users of a system.

Quality-of-service (non-functional) requirements

are requirements that do not perform a specific function for the business requirement but are needed to support the functionality. A few examples are (not an exhaustive list): performance, scalability, security and usability. These are often included within the System Requirements or Functional requirements, where applicable.

Implementation (transition) requirements

are capabilities or behaviors required only to enable transition from the current state of the enterprise to the desired future state, but that will thereafter no longer be required.

Report specifications

define the purpose of a report, its justification, attributes and columns, owners and runtime parameters.

The traceability matrix

is a cross matrix for recording the requirements through each stage of the requirements gathering process. High level concepts will be matched to scope items which will map to individual requirements which will map to corresponding functions. This matrix should also take into account any changes in scope during the life of

the project. At the end of a project, this matrix should show each function built into a system, its source and the reason that any stated requirements may not have been delivered.

Within the systems development life cycle domain (SDLC), the business analyst typically performs a liaison function between the business side of an enterprise and the providers of services to the enterprise. A common alternative role in the IT sector is business analyst, systems analyst, and functional analyst, although some organizations may differentiate between these titles and corresponding responsibilities.

Prerequisites

There is no defined way to become a business analyst. Often the BA has a technical background, whether having worked as a programmer or engineer, or completing a Computer Science degree. Others may move into a BA role from a business role – their status as a subject matter expert and their analytical skills make them suitable for the role. Business analysts may overlap into roles such as project manager or consultant. When focused on specific systems, the term Business Systems Analyst may be used.

A BA does not always work in IT-related projects, as BA skills are often required in marketing and financial roles as well.

The International Institute of Business Analysis provides a certification program for business analysts (Certified Business Analyst Professional or CBAP), as well as providing a body of knowledge for the field (Business Analysis Body of Knowledge or BABOK). However this is not well established yet in the Industry (not a requirement to be certified in order to work as a BA).

A few consulting companies provide BA training courses and there are some consulting books on the market (UML, workshop facilitation, consultancy, communication skills). Some helpful text books are:

- *Customer-Centered Products* by Ivy F. Hooks and Kristin A. Farry (Amazon, USA, 2001).
- *UML for the IT Business Analyst: A Practical Guide to Object-Oriented Requirements Gathering* by Howard Podeswa,
- *Writing Effective Use Cases* by Alistair Cockburn
- *Discovering Real Business Requirements for Software Project Success* by Robin F. Goldsmith.
- *Business Modeling with UML* by Eriksson & Penker
- *Software Requirements, 2nd Edition* by Karl E. Wiegers (Microsoft Press, 2003)
- *Stand Back and Deliver: Accelerating Business Agility* by Pollyanna Pixton, Niel Nickolaisen, Todd Little, and Kent McDonald (Addison-Wesley Professional, 2009)

BAs work in different industries such as finance, banking, insurance, telecoms, utilities, software services, and so on. Due to working on projects at a fairly high level of abstraction, BAs can switch between industries. The business domain subject areas BAs may work in include workflow, billing, mediation, provisioning and customer relationship management. The telecom industry has mapped these functional areas in their Telecommunications Operational Map (eTOM) model, Banking in the Interface Frame Work (IFW) and Emergency agencies in the Prevention Preparation Response and Recovery model (PPRR).

Finally, Business Analysts do not have a predefined and fixed role, as they can take a shape in operations (technology architect or project management) scaling, sales planning, strategy devising or even in developmental process. Hence, they get a different name for the played role. Even the International Institute of Business Analysis and its associates have had several editions of the roles and responsibilities of a person undertaking the BA role hence the certification is not as strong as for instance for Project managers PMBOK.

Possible benefits and drawbacks of including business analysts in software projects

The role of the BA is key in software development projects, especially at the inception of the project. They serve as the mediator or the bridge between the Technical and Business end of stakeholders. Typically, in organizations where no formal structure or processes exist, the Business Owners and Developers communicate directly. Assuming that Developers have no organizational skills or time to spend attention to this topic, this can present a problem: the goal of the Business Owner is to get what they want very quickly, and the goal of the Developer is to give the Business Owner what they want as quickly as he/she can give it to him/her. This can lead to creating changes in a vacuum, not necessarily taking the needs of all users of the system into account, depending on the organizational knowledge skills of the involved Developers.

This can lead to a situation where there are no or rarely any detailed definition of the requirements, and many times, the real reason for the request may not make good business sense. There tends to be no emphasis on long term, strategic goals that the business wants to achieve via Information Technology. The Business Analyst can bring structure (i.e. Methodology to gather requirements: As-Is process baseline, To-Be process, workshops) and formalization of requirements (i.e. gather the required Capabilities) into this process, which may lead to increased foresight or outcome among Business Owners. It is very helpful in business development.[3]

Drawbacks include situations where the Business Analyst just works as 'man in the middle', without helping Business Owner, Subject Matter experts and Developers to streamline the long term goals results in a loss of time as well as information.[4]

In recent years, there has been an upsurge of using analysts of all sorts: business analysts, business process analysts, risk analysts, system analysts. Ultimately, an effective project manager will include Business Analysts who break down communication barriers between stakeholders and developers.[5]

References

[1] Paul, Debra; Donald Yeates, Keith Hindle (April 2006). *Business analysis* (http://www.bcs.org/server.php?show=nav.7094). BCS. p. 4. ISBN 978-1-902505-70-1. .

[2] IIBA (March 2009). Kevin Brennan. ed. *A Guide to the Business Analysis Body of Knowledge* (http://www.theiiba.org/AM/Template.cfm?Section=Body_of_Knowledge) (2.0 ed.). IIBA. p. 3. ISBN 978-0-9811292-1-1. .

[3] "Business analyst role key in SOA software development projects" (http://searchsoa.techtarget.com/news/article/0,289142,sid26_gci1325388,00.html). TechTarget. . Retrieved 2008-08-26.

[4] "Business analyst role key in SOA software development projects" (http://searchsoa.techtarget.com/news/article/0,289142,sid26_gci1325388,00.html). TechTarget. . Retrieved 2008-08-26.

[5] "Rethinking the Role of Business Analysts" (http://www.agilemodeling.com/essays/businessAnalysts.htm). . Retrieved 2008-08-26.

External links

- Business analysis and the project lifecycle in practice – a briefing study guide (http://www.cilco.co.uk/briefing-studies/index.html)

Systems analysis

Systems analysis is the study of sets of interacting entities, including computer systems analysis. This field is closely related to requirements analysis or operations research. It is also "an explicit formal inquiry carried out to help someone (referred to as the decision maker) identify a better course of action and make a better decision than he might otherwise have made."[1]

Overview

The terms analysis and synthesis come from Greek where they mean respectively "to take apart" and "to put together". These terms are in scientific disciplines from mathematics and logic to economics and psychology to denote similar investigative procedures. Analysis is defined as the procedure by which we break down an intellectual or substantial whole into parts. Synthesis is defined as the procedure by which we combine separate elements or components in order to form a coherent whole.[2] Systems analysis researchers apply methodology to the analysis of systems involved to form an overall picture. System analysis is used in every field where there is a work of developing something. Analysis can also be defined as a series of components that perform organic function together.

Information technology

The development of a computer-based information system includes a systems analysis phase which produces or enhances the data model which itself is a precursor to creating or enhancing a database (see Christopher J. Date "An Introduction to Database Systems"). There are a number of different approaches to system analysis. When a computer-based information system is developed, systems analysis (according to the Waterfall model) would constitute the following steps:

- The development of a feasibility study, involving determining whether a project is economically, socially, technologically and organizationally feasible.
- Conducting fact-finding measures, designed to ascertain the requirements of the system's end-users. These typically span interviews, questionnaires, or visual observations of work on the existing system.
- Gauging how the end-users would operate the system (in terms of general experience in using computer hardware or software), what the system would be used for and so on

Another view outlines a phased approach to the process. This approach breaks systems analysis into 5 phases:

- Scope Definition
- Problem analysis
- Requirements analysis
- Logical design
- Decision analysis

Use cases are a widely-used systems analysis modeling tool for identifying and expressing the functional requirements of a system. Each use case is a business scenario or event for which the system must provide a defined response. Use cases evolved out of object-oriented analysis; however, their use as a modeling tool has become common in many other methodologies for system analysis and design.

Practitioners

Practitioners of systems analysis are often called up to dissect systems that have grown haphazardly to determine the current components of the system. This was shown during the year 2000 re-engineering effort as business and manufacturing processes were examined as part of the Y2K automation upgrades.[3] Employment utilizing systems analysis include systems analyst, business analyst, manufacturing engineer, enterprise architect, etc.

While practitioners of systems analysis can be called upon to create new systems, they often modify, expand or document existing systems (processes, procedures and methods). A set of components interact with each other to accomplish some specific purpose. Systems are all around us. Our body is itself a system. A business is also a system. People, money, machine, market and material are the components of business system that work together that achieve the common goal of the organization.

References

[1] SYSTEMS ANALYSIS (http://web.archive.org/web/20070822025602/http://pespmc1.vub.ac.be/ASC/SYSTEM_ANALY.html)
[2] Tom Ritchey, [http://www.swemorph.com/pdf/anaeng-r.pdf Analysis and .
[3] Géza HUSI: Mechatronics Control Systems

External links

- Systems Analysis, Modelling and Prediction (SAMP), University of Oxford (http://www.eng.ox.ac.uk/samp)
- Software Requirement Analysis using UML (http://www.slideshare.net/dhirajmusings/software-requirement-analysis-using-uml) article by Dhiraj Shetty.
- *Introduction to Social Macrodynamics* (http://urss.ru/cgi-bin/db.pl?cp=&page=Book&id=34250&lang=en&blang=en&list=Found)
- A useful set of guides and a case study about the practical application of business and systems analysis methods (http://www.cilco.co.uk/briefing-studies/index.html)
- Complete online tutorial for system analysis and design (http://www.systemsanalysis.co.nr)
- A comprehensive description of the discipline of systems analysis from Simmons College, Boston, MA, USA (www.simmons.edu) (http://web.simmons.edu/~benoit/LIS486/SystemsAnalysis.html)

Information architecture

Information architecture (**IA**) is the art and science of organizing and labelling websites, intranets, online communities and software to support usability.[1] It is an emerging discipline and *community of practice* focused on bringing together principles of design and architecture to the *digital landscape*.[2] Typically it involves a model or concept of information which is used and applied to activities that require explicit details of complex information systems. These activities include library systems and database development.

Historically the term "information architect" is attributed to Richard Saul Wurman,"[3] and now there is a growing network of active IA specialists known as the Information Architecture Institute.[4]

Definition

Information architecture has somewhat different meanings in different branches of IS or IT architecture. Most definitions have common qualities: a structural design of shared environments, methods of organizing and labeling websites, intranets, and online communities, and ways of bringing the principles of design and architecture to the digital landscape.

Example definitions include:

1. The structural design of shared information environments.[2]
2. The art and science of organizing and labeling web sites, intranets, online communities, and software to support findability and usability.[5][2]
3. An emerging community of practice focused on bringing principles of design and architecture to the digital landscape.[2]
4. The combination of organization, labeling, search and navigation systems within websites and intranets.[2]
5. An emerging discipline and community of practice focused on bringing principles of design and architecture to the digital landscape.[2]

Debate

The difficulty in establishing a common definition for "information architecture" arises partly from the term's existence in multiple fields. In the field of systems design, for example, information architecture is a component of enterprise architecture that deals with the information component when describing the structure of an enterprise.

While the definition of information architecture is relatively well-established in the field of systems design, it is much more debatable within the context of online information systems (i.e., websites). Andrew Dillon refers to the latter as the "big IA-little IA debate"[6]. In the little IA view, information architecture is essentially the application of information science to webdesign, which considers, for example, issues of classification and information retrieval. In the big IA view, information architecture involves more than just the organization of a website; it also factors in user experience, thereby considering usability issues of information design.

The role of IA

Information architecture is a specialized skill set that interprets information and expresses distinctions between signs and systems of signs. More concretely, it involves the categorization of information into a coherent structure, preferably one that the intended audience can understand quickly, if not inherently, and then easily retrieve the information for which they are searching[2]. The organization structure is usually hierarchical, but can have other structures, such as concentric or even chaotic[2]. Typically this is required in activities such as library systems, Content Management Systems, web development, user interactions, database development, programming, technical writing, enterprise architecture, and critical system software design. Information architecture originates, to some degree, in the library sciences. Many schools with library and information science departments teach information

architecture.[7]

In the context of information systems design, information architecture refers to the analysis and design of the data stored by information systems, concentrating on entities, their attributes, and their interrelationships. It refers to the modeling of data for an individual database and to the corporate data models an enterprise uses to coordinate the definition of data in several (perhaps scores or hundreds) distinct databases. The "canonical data model" is applied to integration technologies as a definition for specific data passed between the systems of an enterprise. At a higher level of abstraction it may also refer to the definition of data stores.

Information architect

Richard Saul Wurman says of the term information architect "used in the words architect of foreign policy. I mean architect as in the creating of systemic, structural, and orderly principles to make something work--the thoughtful making of either artifact, or idea, or policy that informs because it is clear."[8]

References

[1] 'What is IA?' Information Architecture Institute. IAinstitute.org (http://www.iainstitute.org/documents/learn/What_is_IA.pdf)
[2] Rosenfeld, L., Morville, P. (1998). Information architecture for the World Wide Web. 1 st. Ed., O'Reilly & Associates, Sebastopol, CA.
[3] R.S. Wurman: "Information Architects"
[4] 'Join the IA Network' Information Architecture Institute. (http://www.iainstitute.org/en/network/)
[5] 'What is IA?' Information Architecture Institute. IAinstitute.org (http://www.iainstitute.org/documents/learn/What_is_IA.pdf)
[6] Dillon, A. (2002) Information Architecture in JASIST: Just where did we come from? Journal of the American Society for Information Science and Technology, 53(10), 821-823.
[7] IAinstitute.org (http://www.iainstitute.org/en/learn/education/schools_teaching_ia.php), Schools Teaching IA
[8] R.S. Wurman: "Information Architects"

Further reading

- Peter Morville and Louis Rosenfeld, *Information Architecture for the World Wide Web* (http://books.google.com/books?id=2d2Ry2hZc2MC&printsec=frontcover&dq=information+architecture&hl=en&ei=jXxyTc-6MpHCvgOF0eS9AQ&sa=X&oi=book_result&ct=result&resnum=1&ved=0CCcQ6AEwAA#v=onepage&q&f=false) (2006), ISBN 0-596-52734-9
- Wei Ding; Xia Lin (15 May 2009). *Information Architecture: The Design and Integration of Information Spaces* (http://books.google.com/books?id=-wy3RhKoWWQC). Morgan & Claypool Publishers. ISBN 978-1-59829-959-5.

Solutions Architect

A **Solutions Architect** in Information Technology Enterprise Architecture is a practitioner in the field of Solution Architecture.[1]

The role title has a wider meaning in relation to solving problems, but is more often used in the narrower domain of Technical architecture - the context for the remainder of this definition. In this context, the Solutions Architect is a very experienced architect with cross-domain, cross-functional and cross-industry expertise. He/she outlines solution architecture descriptions, then monitors and governs their implementation.

Overview of solutions architect

The role of "Solutions Architect" requires the knowledge and skills that are both broad and deep. To be effective the Solutions Architect must have experience on multiple Hardware and Software Environments and be comfortable with complex heterogeneous systems environments. The Solutions Architect is often a highly seasoned senior technocrat who has led multiple projects through the Software development process or Systems Development Life Cycle (SDLC), and has usually performed in a variety of different roles in that life cycle. The person needs an ability to share and communicate ideas verbally, both orally and in writing, to executive staff, business sponsors, and technical resources in clear concise language that is the parlance of each group.

A practitioner of Solution Architecture, Systems engineering and Software engineering processes, the Solutions Architect is the person who organizes the development effort of a systems solution. The Solutions Architect is responsible for the development of the overall vision that underlies the projected solution and transforms that vision through execution into the solution. The Solutions Architect becomes involved with a project at the time of inception and is involved in the Functional analysis (FA) of developing the initial requirements. They then remain involved throughout the balance of the project.

The Solutions Architect is an expert in many categories. They should have hands-on experience in multiple industries and across several disciplines. They can master a variety of hardware platforms including mainframes, distributed platforms, desktops, and mobile devices. Akin to that they should also possess skill and understanding of a variety of Operating Systems. A broad and deep understanding of Databases is also required.

Solutions Architects decide which technologies to use. They work very closely with developers to ensure proper implementation. They are the link between the needs of the organization and the developers.

Positioning solution architects in relation to enterprise architects

Solution Architects in large organizations often act as the bridge between Enterprise Architects and Application Architects.

An enterprise architect's deliverables are usually more abstract than a solution architect's deliverables. But that is not always the case, and the main distinction between enterprise architect and solution architect lies in their different motivations. The enterprise architect is primarily employed in the design, planning and governance of strategic and cross-organisational rationalisation or optimisation of an enterprise's services, processes or components. The solution architect is primarily employed to help programme and project managers in the design, planning and governance of implementation projects of any kind.

A solution architect may have a reporting line to an enterprise architect, but the influence the enterprise architect team has on solution architects depends on an organisation's policies and management structure. So, the extent to which a solution architect's work derives from enterprise architects' road maps will vary from 0 to 100 percent.

When the solution architect starts and stops depends on the funding model for the process of solution identification and delivery. E.g. An enterprise may employ a solution architect on a feasibility study, or to prepare a solution vision or solution outline for an Invitation to Tender. A supplier may employ a solution architect at "bid time", before any

implementation project is costed or resourced. Both may employ a solution architect to govern an implementation project, or play a leading role within it.

An IT services provider may employ a solution architect in a role that reports to a senior architect who is: 1. focused on operational services rather than implementation programme/projects, where understanding managed operations is important. 2. responsible for coordinating all services provided to one organisation by way of strategy, business consulting, projects and operational services. 3. working on a bid to supply one organisation with all the services above, or a framework bid that covers more than one customer organization at a more strategic level. In cases 2 and 3, the senior architect is a kin to an enterprise architect, but (in the UK at least) is more likely to be called Solution Director, Service Director, Technical Director or CTO.

The Solutions Architect has several essential duties and responsibilities, which include all or some combination of the following:

Solutions Architect topics

Business Planning and General Management

- Take ownership of a particular solution offering
- Develop and execute a solution strategy and business plan that support product growth
- Shape, design, and plan specific service lines in product area
- Spearhead product marketing

Subject Matter Expertise

- Act as visionary and strategist for solution product area
- Survey market landscape for solution insights, direction, vendors, and methods
- Provides expertise to identify and translate system requirements into software design documentation,
- Work with technical writers to ensure quality internal and external client-oriented documentation
- Speak at trade conferences and seek authorship opportunities in trade publications

Business Development

- Help marketing departments develop marketing materials and position strategies for product area, in conjunction with overall marketing message framework
- Help business development life cycle by serving as a product SME to help identify and qualify business development opportunities
- Manages sales and marketing activities for the service offering
- With Channel Development team, develop and maintain vendor relationships within the product

Methodology and Quality Assurance

- Lead development of formalized solution methodologies
- Build and maintain repository for deliverables, methodologies, and business development documents
- Interface and coordinate tasks with internal and external technical resources. Collaborates with Project Managers and technical directors to provision estimates, develop overall implementation solution plan, and serve lead as required, to implement installation, customization, and integration efforts
- Oversee aspects of project life cycle, from initial kickoff through requirements analysis, design and implementation phases for projects within solution area
- Provide quality assurance for services within solution area
- Write, or direct the writing of white papers that further insight and thought in the solution area

Work Force Management, Supervision and Mentoring

- Manages a team of direct reports who drive service lines in the solution area
- Assists staffing coordinators who define project team requirements for projects in solution area
- Work with Delivery Services Director to define overall recruiting needs and expertise in solution area
- Work with Director of Delivery Services to establish professional development needs for practitioners in solution area
- Mentor and guide more junior technical resources

Architect metaphor

The use of any form of the word 'architect' is regulated by 'title acts' in many states in the UK and the US, and a person must be licensed as a building architect to use it.[2]

References

[1] Anatomy of a Software Development Role: Solution Architect (http://www.developer.com/mgmt/article.php/3504496/Anatomy-of-a-Software-Development-Role-Solution-Architect.htm), developer.com

[2] The term "architect" is a **professional title** protected by law and restricted, in most of the world's jurisdictions, to those who are trained in the planning, design and supervision of the construction of **buildings**. In these jurisdictions, anyone who is not a licensed architect is prohibited from using this title *in any way*. In the State of New York, and in other U.S. states, the unauthorized use of the title "architect" is a crime and is subject to criminal proceedings. "Architecture: What's Legal, What's Not" (http://www.aianys.org/what's_legal_whats_not.pdf). AIA New York State. . Retrieved 9 Jul 2012. "NYS Architecture:Laws, Rules & Regulations:Article 147 Architecture" (http://www.op.nysed.gov/prof/arch/article147.htm). . Retrieved 9 Jul 2012.

Software architect

Software architect is a computer programmer that makes high-level design choices and dictates technical standards, including software coding standards, tools, or platforms.

History

With the increasing popularity of multi-tier application development, the choices of how an application can be built have also increased. Given that expansion, the risk that a software development project may inadvertently create a "new" end product that, in essence, already existed has grown markedly. A new 'Software architect' role became necessary during software development .

The software architect concept began to take hold when object oriented programming (OOP) was coming into more widespread use (in the late 1990s and early years of the 21st century) . OOP allowed ever-larger and more complex applications to be built, which in turn required increased high-level application and system oversight.

The main responsibilities of a software architect include:

- Limiting choices available during development by
  - choosing a standard way of pursuing application development
  - creating, defining, or choosing an application framework for the application
- Recognizing potential reuse in the organization or in the application by
  - Observing and understanding the broader system environment
  - Creating the component design
  - Having knowledge of other applications in the organization
- Subdivide a complex application, during the design phase, into smaller, more manageable pieces
- Grasp the functions of each component within the application

- Understand the interactions and dependencies among components
- Communicate these concepts to developers

In order to perform these responsibilities effectively, software architects often use tools or standardized model and symbol sets such as Unified Modeling Language and OOP to represent systems or develop artifacts. UML has become an important tool for software architects to use in communicating the overall system design to developers and other team members, comparable to the drawings made by building architects.

Duties

The role of software architect generally has certain common traits:

Architect makes high-level design choices much more often than low-level choices. In addition, the architect may sometimes dictate technical standards, including coding standards, tools, or platforms.

Software architects may also be engaged in the design of the architecture of the hardware environment, or may focus entirely on the design methodology of the code.

Architects can use various software architectural models that specialize in communicating architecture.

Other types of IT-related Architects

The enterprise architect handles the interaction between the business and IT sides of an organization and is principally involved with determining the AS-IS and TO-BE states from a business and IT process perspective. Unfortunately many organizations are bundling the software architect duties within the role of Enterprise Architecture. This is primarily done as an effort to "up-sell" the role of a software architect and/or to merge two disparate business-related disciplines to avoid overhead.

An application architect works with a single software application. This may be a full- or a part-time role. The application architect is almost always an active software developer .

Other similar titles in use, but without consensus on their exact meaning, include:

- Solutions Architect, which may refer to a person directly involved in advancing a particular business solution needing interactions between multiple applications. May also refer to an Application Architect.
- System Architect (singular), which is often used as a synonym for Application Architect. However, if one subscribes to Systems theory and the idea that an enterprise can be a system, then System Architect could also mean Enterprise Architect.
- Systems Architect (plural), which is often used as a synonym for Enterprise Architect or Solutions Architect.

The table below indicates many of the differences between various kinds of software architects:

Architect Type	Strategic Thinking	System Interactions	Communication	Design
Enterprise Architect	Across Projects	Highly Abstracted	Across Organization	Minimal, High Level
Solutions Architect	Focused on solution	Very Detailed	Multiple Teams	Detailed
Application Architect	Component re-use, maintainability	Centered on single Application	Single Project	Very Detailed

In the software industry, as the table above suggests, the various versions of architect do not always have the same goals.[1]

References

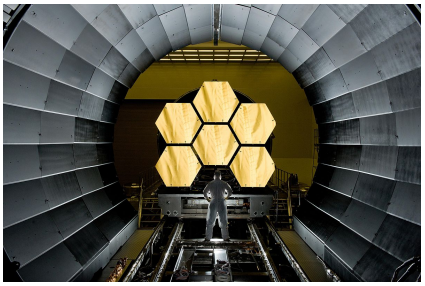
[1] Anecdote about an interaction between Solution and Enterprise Architect (http://blogs.tedneward.com/2007/09/20/Hard+Questions+About+Architects.aspx)

External links

- International Association of Software Architects (IASA) (http://www.iasahome.org/)

Systems architect

Systems Architect



Systems Architects divide large and complex systems into manageable subsystems that can be handled by individual engineers.

Occupation	
Names	Systems Architect
Activity sectors	Systems Engineering Systems Design Engineering
Description	
Competencies	scientific knowledge, engineering, planning and management skills
Education required	see professional requirements

The ***systems architect*** establishes the basic structure of the system, defining the essential core design features and elements that provide the framework for all that follows, and are the hardest to change later. The systems architect provides the architects view of the users' vision for what the system needs to be and do, and the paths along which it must be able to evolve, and strives to maintain the integrity of that vision as it evolves during detailed design and implementation.

Overview

In systems design, the architects and engineers are responsible for:

- Interfacing with the user(s) and sponsor(s) and all other stakeholders in order to determine their (evolving) needs.
- Generating the highest level of system requirements, based on the user's needs and other constraints such as cost and schedule.
- Ensuring that this set of high level requirements is consistent, complete, correct, and operationally defined.
- Performing cost-benefit analyses to determine whether requirements are best met by manual, software, or hardware functions; making maximum use of commercial off-the-shelf or already developed components.
- Developing partitioning algorithms (and other processes) to allocate all present and foreseeable requirements into discrete partitions such that a minimum of communications is needed among partitions, and between the user and the system.
- Partitioning large systems into (successive layers of) subsystems and components each of which can be handled by a single engineer or team of engineers or subordinate architect.
- Interfacing with the design and implementation engineers and architects, so that any problems arising during design or implementation can be resolved in accordance with the fundamental design concepts, and user needs and constraints.
- Ensuring that a maximally robust design is developed.

- Generating a set of acceptance test requirements, together with the designers, test engineers, and the user, which determine that all of the high level requirements have been met, especially for the computer-human-interface.
- Generating products such as sketches, models, an early user guide, and prototypes to keep the user and the engineers constantly up to date and in agreement on the system to be provided as it is evolving.
- Ensuring that all architectural products and products with architectural input are maintained in the most current state and never allowed to become obsolete.

Systems architect: topics

Large systems architecture was developed as a way to handle systems too large for one person to conceive of, let alone design. Systems of this size are rapidly becoming the norm, so architectural approaches and architects are increasingly needed to solve the problems of large systems.

Users and sponsors

Architects *are* expected to understand human needs and develop humanly functional and aesthetically pleasing products. A good architect is also the principal keeper of the user's vision of the end product— and of the process of deriving requirements from and implementing that vision.

An architect does not follow an exact procedure. S/he communicates with users/sponsors in a highly interactive way— together they extract the true *requirements* necessary for the designed system. The architect must remain constantly in communication with the end users. Therefore, the architect must be intimately familiar with the user's environment and problem.

High level requirements

The user requirements' specification should be a joint product of the user and designer: the user brings his needs and wish list, the architect brings knowledge of what is likely to prove doable within cost and time constraints. When the user needs are translated into a set of high level requirements is also the best time to write the first version of the acceptance test, which should, thereafter, be religiously kept up to date with the requirements. That way, the user will be absolutely clear about what s/he is getting. It is also a safeguard against untestable requirements, misunderstandings, and requirements creep.

The development of the first level of engineering requirements is not a purely analytical exercise and should also involve both the architect and engineer. If any compromises are to be made— to meet constraints like cost, schedule, power, or space, the architect must ensure that the final product and overall look and feel do not stray very far from the user's intent. The engineer should focus on developing a design that optimizes the constraints but ensures a workable and reliable product. The provision of needed services to the user is the true function of an engineered system. However, as systems become ever larger and more complex, and as their emphases move away from simple hardware and software components, the narrow application of traditional systems development principles is found to be insufficient— the application of the more general principles of systems, hardware, and software architecture to the design of (sub)systems is seen to be needed. An architecture is also a simplified model of the finished end product— its primary function is to define the parts and their relationships to each other so that the whole can be seen to be a consistent, complete, and correct representation of what the user had in mind— especially for the computer-human-interface. It is also used to ensure that the parts fit together and relate in the desired way.

It is necessary to distinguish between the architecture of the user's world and the engineered systems architecture. The former represents and addresses problems and solutions in the *user's* world. It is principally captured in the computer-human-interfaces (CHI) of the engineered system. The engineered system represents the *engineering* solutions— how the *engineer* proposes to develop and/or select and combine the components of the technical infrastructure to support the CHI. In the absence of an experienced architect, there is an unfortunate tendency to confuse the two architectures. But— the engineer thinks in terms of hardware and software and the technical solution

space, whereas the user may be thinking in terms of solving a problem of getting people from point A to point B in a reasonable amount of time and with a reasonable expenditure of energy, or of getting needed information to customers and staff. A systems architect is expected to combine knowledge of both the architecture of the user's world and of (all potentially useful) engineering systems architectures. The former is a joint activity with the user; the latter is a joint activity with the engineers. The product is a set of high level requirements reflecting the user's requirements which can be used by the engineers to develop systems design requirements.

Because requirements evolve over the course of a project, especially a long one, an architect is needed until the system is accepted by the user: the architect insures that all changes and interpretations made during the course of development do not compromise the user's viewpoint.

Cost/benefit analyses

Architects are generalists. They are not expected to be experts in any one technology but are expected to be knowledgeable of many technologies and able to judge their applicability to specific situations. They also apply their knowledge to practical situations, but evaluate the cost/benefits of various solutions using different technologies, for example, hardware versus software versus manual, and assure that the system as a whole performs according to the user's expectations.

Many commercial-off-the-shelf or already developed hardware and software components may be selected independently according to constraints such as cost, response, throughput, etc. In some cases, the architect can already assemble the end system unaided. Or, s/he may still need the help of a hardware or software engineer to select components and to design and build any special purpose function. The architects (or engineers) may also enlist the aid of specialists— in safety, security, communications, special purpose hardware, graphics, human factors, test and evaluation, quality control, RMA, interface management, etc. An effective systems architectural team must have immediate access to specialists in critical specialties.,

Partitioning and layering

An architect planning a building works on the overall design, making sure it will be pleasing and useful to its inhabitants. While a single architect by himself may be enough to build a single-family house, many engineers may be needed, in addition, to solve the detailed problems that arise when a novel high-rise building is designed. If the job is large and complex enough, parts of the architecture may be designed as independent components. That is, if we are building a housing complex, we may have one architect for the complex, and one for each type of building, as part of an *architectural team.*

Large automation systems also require an architect and much engineering talent. If the engineered system is large and complex enough, the systems architect may defer to a hardware architect and a software architect for parts of the job, although they all may be members of a joint architectural team.

The architect should sub-allocate the system requirements to major components or subsystems that are within the scope of a single hardware or software engineer, or engineering manager and team. *But the architect must never be viewed as an engineering supervisor.* (If the item is sufficiently large and/or complex, the chief architect will sub-allocate portions to more specialized architects.) Ideally, each such component/subsystem is a sufficiently stand-alone object that it can be tested as a complete component, separate from the whole, using only a simple testbed to supply simulated inputs and record outputs. That is, it is not necessary to know how an air traffic control system works in order to design and build a data management subsystem for it. This is only necessary to know the constraints under which the subsystem will be expected to operate.

A good architect ensures that the system, however complex, is built upon relatively simple and "clean" concepts for each (sub)system or layer and is easily understandable by everyone, especially the user, without special training. The architect will use a minimum of heuristics to ensure that each partition is well defined and clean of kludges, work-arounds, short-cuts, or confusing detail and exceptions. As user needs evolve, (once the system is fielded and

in use), it is a lot easier subsequently to evolve a simple concept than one laden with exceptions, special cases, and lots of "fine print."

Layering the architecture is important for keeping the architecture sufficiently simple at each *layer* so that it remains comprehensible to a single mind. As layers are ascended, whole systems at *lower layers* become simple *components* at the *higher layers,* and may disappear altogether at the *highest layers.*

Acceptance test

The acceptance test is a principal responsibility of the systems architect. It is the chief means by which the project lead will prove to the user that the system is as originally planned and that all involved architects and engineers have met their objectives.

Communications with users and engineers

A building architect uses sketches, models, and drawings. An automation systems (or software or hardware) architect should use sketches, models, and prototypes to discuss different solutions and results with users, engineers, and other architects. An early, draft version of the user's manual is invaluable, especially in conjunction with a prototype. Nevertheless, it is important that a workable, well written *set of requirements,* or specification, be created which is understandable to the customer (so that they can properly sign off on it). But it must use precise and unambiguous language so that designers and other implementers are left in no doubt as to meanings or intentions. In particular, all requirements must be testable, and the initial draft of the test plan should be developed contemporaneously with the requirements. All stakeholders should sign off on the acceptance test descriptions, or equivalent, as the sole determinant of the satisfaction of the requirements, at the outset of the program.

References

Further reading

- Donald Firesmith et al.: *The Method Framework for Engineering System Architectures*, (2008)
- Mark W. Maier and Rechtin, Eberhardt, *The Art of Systems Architecting*, Third Edition (2009)
- Gerrit Muller, "Systems architecting: A business perspective," CRC Press, (2012).
- Eberhardt Rechtin, *Systems Architecting: Creating & Building Complex Systems*, 1991.
- Rob Williams, *Computer Systems Architecture: a Networking Approach*, Second Edition (December, 2006).

External links

- Systems Architecture : Canaxia Brings an Architect on Board (http://www.informit.com/articles/article.aspx?p=169564), Article.

Project manager

A **project manager** is a professional in the field of project management. Project managers can have the responsibility of the planning, execution and closing of any project, typically relating to construction industry, architecture, Aerospace and Defence, computer networking, telecommunications or software development.

Many other fields in the production, design and service industries also have project managers.

US Navy 080111-N-8273J-033 Chief of Naval Operations (CNO) Adm. Gary Roughead talks with project managers

Overview

A project manager is the person responsible for accomplishing the stated project objectives. Key project management responsibilities include creating clear and attainable project objectives, building the project requirements, and managing the constraints of the project management triangle, which are *cost*, *time*, *scope*, and quality.

A project manager is often a client representative and has to determine and implement the exact needs of the client, based on knowledge of the firm they are representing. The ability to adapt to the various internal procedures of the contracting party, and to form close links with the nominated representatives, is essential in ensuring that the key issues of cost, time, quality and above all, client satisfaction, can be realized.

The term and title 'project manager' has come to be used generically to describe anyone given responsibility to complete a project. However, it is more properly used to describe a person with full responsibility and the same level of authority required to complete a project. If a person does not have high levels of both responsibility and authority then they are better described as a project administrator, coordinator, facilitator or expeditor.

Project manager topics

Project management

> Project Management is quite often the province and responsibility of an individual project manager. This individual seldom participates directly in the activities that produce the end result, but rather strives to maintain the progress and mutual interaction and tasks of various parties in such a way that reduces the risk of overall failure, maximizes benefits, and restricts costs.

Products and services

> Any type of product or service — pharmaceuticals, building construction, vehicles, electronics, computer software, financial services, etc. — may have its implementation overseen by a project manager and its operations by a product manager.

Project tools

> The tools, knowledge and techniques for managing projects are often unique to Project Management. For example: work breakdown structures, critical path analysis and earned value management. Understanding and applying the tools and techniques which are generally recognized as good practices are not sufficient alone for effective project management. Effective project management requires that the project manager understands

and uses the knowledge and skills from at least four areas of expertise. Examples are PMBOK, Application Area Knowledge: standards and regulations set forth by ISO for project management, General Management Skills and Project Environment Management[1] There are also many options for project management software to assist in executing projects for the project manager and his/her team.

Project teams

When recruiting and building an effective team, the manager must consider not only the technical skills of each person, but also the critical roles and chemistry between workers. A project team has mainly three separate components: Project Manager, Core Team and Contracted Team.

Risk

Most of the project management issues that influence a project arise from risk, which in turn arises from uncertainty. The successful project manager focuses on this as his/her main concern and attempts to reduce risk significantly, often by adhering to a policy of open communication, ensuring that project participants can voice their opinions and concerns.

Types of project managers

Construction Project Manager

Construction project managers in the past were individuals, who worked in construction or supporting industries and were promoted to foreman. It was not until the late 20th century that construction and Construction management became distinct fields.

Until recently, the American construction industry lacked any level of standardization, with individual States determining the eligibility requirements within their jurisdiction. However, several Trade associations based in the United States have made strides in creating a commonly-accepted set of qualifications and tests to determine a project manager's competency.

- The Project Management Institute has made some headway into being a standardizing body with its creation of the Project Management Professional (PMP) designation.
- The Constructor Certification Commission of the American Institute of Constructors holds semiannual nationwide tests. Eight American Construction Management programs require that students take these exams before they may receive their Bachelor of Science in Construction Management degree, and 15 other Universities actively encourage their students to consider the exams.
- The Associated Colleges of Construction Education, and the Associated Schools of Construction have made considerable progress in developing national standards for construction education programs.

The profession has recently grown to accommodate several dozen Construction Management Bachelor of Science programs.

The United States Navy construction battalions, nicknamed the SeaBees, puts their command through strenuous training and certifications at every level. To become a Chief Petty Officer in the SeaBees is equivalent to a BS in Construction Management with the added benefit of several years of experience to their credit. See ACE accreditation.

Architectural Project Manager

Architectural project manager are project managers in the field of architecture. They have many of the same skills as their counterpart in the construction industry. An architect will often work closely with the construction project manager in the office of the General contractor (GC), and at the same time, coordinate the work of the design team and numerous consultants who contribute to a construction project, and manage communication with the client. The issues of budget, scheduling, and quality-control are the responsibility of the Project Manager in an architect's office.

Software Project Manager

A Software Project Manager has many of the same skills as their counterparts in other industries. Beyond the skills normally associated with traditional project management in industries such as construction and manufacturing, a software project manager will typically have an extensive background in software development. Many software project managers hold a degree in Computer Science, Information Technology or another related field.

In traditional project management a heavyweight, predictive methodology such as the waterfall model is often employed, but software project managers must also be skilled in more lightweight, adaptive methodologies such as DSDM, Scrum and XP. These project management methodologies are based on the uncertainty of developing a new software system and advocate smaller, incremental development cycles. These incremental or iterative cycles are timeboxed (constrained to a known period of time, typically from one to four weeks) and produce a working subset of the entire system deliverable at the end of each iteration. The increasing adoption of lightweight approaches is due largely to the fact that software requirements are very susceptible to change, and it is extremely difficult to illuminate all the potential requirements in a single project phase before the software development commences.

The software project manager is also expected to be familiar with the Software Development Life Cycle (SDLC). This may require in depth knowledge of requirements solicitation, application development, logical and physical database design and networking. This knowledge is typically the result of the aforementioned education and experience. There is not a widely accepted certification for software project managers, but many will hold the Project Management Professional (PMP) designation offered by the Project Management Institute, PRINCE2 or an advanced degree in project management, such as a MSPM or other graduate degree in technology management.

Responsibilities

The project manager is accountable for ensuring that everyone on the team knows and executes his or her role, feels empowered and supported in the role, knows the roles of the other team members and acts upon the belief that those roles will be performed.[2] The specific responsibilities of the Project Manager may vary depending on the industry, the company size, the company maturity, and the company culture. However, there are some responsibilities that are common to all Project Managers, noting[3]:

- Developing the project plan
- Managing the project stakeholders
- Managing the project team
- Managing the project risk
- Managing the project schedule
- Managing the project budget
- Managing the project conflicts

Education, certifications and networks

Individuals wishing to obtain professional certifications may take one or more of the offerings available from a variety of organizations:

The Project Management Institute offers the following credentials to project managers:[4]

- Project Management Professional (PMP)
- Certified Associate in Project Management (CAPM),
- Program Management Professional (PgMP)
- PMI Risk Management Professional (PMI-RMP), and
- PMI Scheduling Professional (PMI-SP)

Other institutions and organizations:

- The University of Wisconsin's Masters Certificate in Project Management [5]
- CompTIA offers Project+ Certification
- The Canadian Construction Association (CCA) offers Gold Seal Certification as a Project Manager. [6]
- The UK Office of Government Commerce offers PRINCE2 certification.
- The Australian Institute of Project Management (AIPM) offers Registered Project Manager (RegPM) certification.
- The Defense Acquisition University (DAU) and its School of Program Management offers practitioner training in every element of project management for members of the Federal Government, Defense industry and allied nations.

There are other graduate degrees in project and technology management, such as an MSPM. However, the majority of all project management skills may be developed through the completion of a Ph.D, D.Phil or other similar higher Doctorate.

The IPMA is an international network of national project management societies such as Association for Project Management in the UK. IPMA serves as an umbrella organisation representing national societies which offer their certifications.

Project Management training

Methods of Project Management training are very diverse. Much of the training received by most project managers is on the job training. Other sources of training include

- University degree programs in project management
- Business degree programs with some level of Project Management emphasis
- Certification preparatory classes and training
- Social media such as blogs and podcasts
- Books
- Seminars and conferences
- Local group meetings (I.E. local chapters)

References

[1] PMBOK Guide Third Edition 2004 p.12

[2] Russell, PMP, D. (2011). Accountability. In Succeeding in the project management jungle: How to manage the people side of projects (p. 29). New York, NY: AMACOM.

[3] Berrie, Michele, Project Manager Responsibilities (http://www.pmhut.com/project-manager-responsibilities), PM Hut. Accessed 17. Oct 2009.

[4] Project Management Institute Family of Credentials (http://www.pmi.org/CareerDevelopment/Pages/AboutPMIsCredentials.aspx)

[5] http://exed.wisc.edu/cert/projectmanagement/info.asp

[6] http://www.goldsealcertification.com/overview/about/about_e.asp

Further reading

- US DoD (2003). *Interpretive Guidance for Project Manager Positions* (http://www.opm.gov/fedclass/cg03-0001.pdf). August 2003.
- Open source handbook for project managers *Open source handbook for project managers* (http://www.projectmanagement-training.net/book/). July 2006.

Project management office

A Project Management Office (PMO) is a group or department within a business, agency or enterprise that defines and maintains standards for project management within the organization. The PMO strives to standardize and introduce economies of repetition in the execution of projects. The PMO is the source of documentation, guidance and metrics on the practice of project management and execution. In some organisations this is known as the **Program Management Office** (sometimes abbreviated to **PgMO** to differentiate); the subtle difference is that program management relates to governing the management of several related projects.

Traditional PMOs base project management principles on industry-standard methodologies such as PMBOK or PRINCE2. Increasingly influential industry certification programs such as ISO9000 and the Malcolm Baldrige National Quality Award (MBNQA) as well as government regulatory requirements such as Sarbanes-Oxley have propelled organizations to standardize processes. Organizations around the globe are defining, borrowing and collecting best practices in process and project management and are increasingly assigning the PMO to exert overall influence and evolution of thought to continual organizational improvement.

According to the Standish CHAOS Report (2009), 68% of projects do not meet time/cost/scope targets. Only 32% of projects were completed on time, within budget and delivered measurable business and stakeholder benefits. There are many reasons for such failures. As per a KPMG survey of 252 organizations, technology is not the most critical factor. Inadequate project management implementation constitutes 32% of project failures, lack of communication constitutes 20% and unfamiliarity with scope and complexity constitutes 17%. Accordingly 69% of project failures are due to lack and/or improper implementation of project management methodologies.

Establishing a PMO group is not a short term strategy to lower costs.[1] Surveys with companies indicates that the longer they have an operating PMO group the better the results achieved to accomplish project goals (which might lead to lowering costs).

PMOs may take other functions beyond standards and methodology, and participate in Strategic Project Management either as facilitator or actively as owner of the Portfolio Management process. Tasks may include Monitoring and Reporting on active projects (following up project until completion), and reporting progress to top management for strategic decisions on what projects to continue or cancel.

A PMO can be one of three types from an organizational exposure perspective: enterprise PMO, organizational (departmental) PMO, or special–purpose PMO. The Project Management Institute (PMI) Program Management Office Community of Practice (CoP), views the PMO as a strategic driver for organizational excellence and seeks to enhance the practices of execution management, organizational governance, and strategic change leadership. As the largest community devoted to the PMO, with over 4,000 members globally, the PMO CoP is the central forum to collaborate, expand the knowledge base, and mature the PMO practice within their own organizations and the business community at large.

References

[1] CIO.com article (http://www.cio.com/article/29887/Why_You_Need_a_Project_Management_Office_PMO_?page=1)

External links

- Project Management Office Community of Practice (http://pmo.vc.pmi.org)
- Project Management Office Processes (http://www.artechhouse.com/GetBlob.aspx?strName=mir-ch03.pdf)
- A compiled library of PMO articles (http://www.pmhut.com/category/project-management-office/)

Chief information officer

Chief information officer (**CIO**), or **information technology** (**IT**) **director**, is a job title commonly given to the most senior executive in an enterprise responsible for the information technology and computer systems that support enterprise goals. The title of Chief Information Officer in Higher Education may be the highest ranking technology executive although depending on the institution, alternative titles are used to represent this position. Generally, the CIO reports to the chief executive officer, chief operations officer or chief financial officer. In military organizations, they report to the commanding officer.

CIO

Information technology and its systems have become so important that the CIO has come to be viewed in many organizations as the key contributor in formulating strategic goals for an organization. The CIO manages the implementation of the useful technology to increase information accessibility and integrated systems management. As a comparison, where the CIO adapts systems through the use of existing technologies, chief technology officer develops new technologies to expand corporate technological capabilities. When both positions are present in an organization, the CIO is generally responsible for processes and practices supporting the flow of information, whereas the CTO is generally responsible for technology infrastructure.

CIO magazine's "State of the CIO 2008" survey asked 558 IT leaders who they report to. The results were: CEO (41%), CFO (23%), COO (16%), Corporate CIO (7%) and Other (13%).[1]

Information technology

The prominence of the CIO position has risen greatly as information, and the information technology that drives it, has become an increasingly important part of the modern organization. The CIO may be a member of the executive committee of an organization, and/or may often be required to engage at board level depending on the nature of the organization and its operating structure and governance environment. No specific qualification are intrinsic of the CIO position, though the typical candidate may have expertise in a number of technological fields - computer science, software engineering, or information systems. Many candidates have Master of Business Administration or Master of Science in Management degrees.[2] More recently CIOs' leadership capabilities, business acumen and strategic perspectives have taken precedence over technical skills. It is now quite common for CIOs to be appointed from the business side of the organization, especially if they have project management skills.

In 2007 a survey amongst CIOs by *CIO* magazine in the UK discovered that their top 10 concerns were: people leadership, managing budgets, business alignment, infrastructure refresh, security, compliance, resource management, managing customers, managing change and board politics.[3]

In 2010, Gartner Executive Programs conducted a global CIO survey and received responses from 2,014 CIOs from 50 countries and 38 industries. [4] Gartner reported that the survey results indicated that the top ten technology priorities for CIOs for 2011 were cloud computing, virtualization, mobile technologies, IT management, business intelligence, networking, voice and data communications, enterprise applications, collaboration technologies, infrastructure, and Web 2.0.

Typically, a CIO is involved with driving the analysis and re-engineering of existing business processes, identifying and developing the capability to use new tools, reshaping the enterprise's physical infrastructure and network access, and with identifying and exploiting the enterprise's knowledge resources. Many CIOs head the enterprise's efforts to integrate the Internet into both its long-term strategy and its immediate business plans. CIO's are often tasked with either driving or heading up crucial IT projects which are essential to the strategic and operational objectives of an organisation. A good example of this would be the implementation of an Enterprise Resource Planning (ERP) system which typically has wide-ranging implications for most organizations. The CIO is evolving into a role where he/she

is creating and monitoring business value from IT assets, to the point where corporate strategist Chris Potts suggests in the novel FruITion that the Chief Information Officer (CIO) be replaced with Chief Internal Investments Officer (CIIO).[5]

Another way that the CIO role is changing is an increased focus on service management.[6] As SaaS, IaaS, BPO and other more flexible value delivery techniques are brought into organizations the CIO usually functions as a 3rd party manager for the organization. In essence, a CIO in the modern organization is required to possess business skills and the ability to relate to the organization as a whole, as opposed to being a technological expert with limited functional business expertise. The CIO position is as much about anticipating trends in the market place with regards to technology as it is about ensuring that the business navigates these trends through expert guidance and proper strategic IT planning that is aligned to the corporate strategy of the organization.

References

[1] "State of the CIO 2008 Data Shows CIO Salaries, Influence Rising" (http://www.cio.com/article/147950/_State_of_the_CIO_2008_Data_Shows_CIO_Salaries_Influence_Rising). CIO. . Retrieved 27 February 2010.

[2] Meridith Levinson (2007-07-05). "Should You Get an MBA? - CIO.com - Business Technology Leadership" (http://www.cio.com/article/122507/Should_You_Get_an_MBA_). CIO.com. . Retrieved 2012-03-28.

[3] "Granger: The final word - CIO UK Magazine" (http://www.cio.co.uk/concern/budgets/features/index.cfm?articleid=351). Cio.co.uk. 2012-03-14. . Retrieved 2012-03-28.

[4] "Gartner Executive Programs Worldwide Survey of More Than 2,000 CIOs Identifies Cloud Computing as Top Technology Priority for CIOs in 2011" (http://www.gartner.com/it/page.jsp?id=1526414). Gartner. . Retrieved 23 March 2011.

[5] fruITion: Creating the Ultimate Corporate Strategy for Information Technology, Chris Potts, Technics Publications, LLC 2008 (http://www.technicspub.com/product.sc?productId=7&categoryId=1)

[6] "CIO Magazine: Recession Shifts IT Service Management into Fast Lane" (http://www.cio.com/article/558564/Recession_Shifts_IT_Service_Management_Into_Fast_Lane). Cio.com. 2010-02-26. . Retrieved 2012-03-28.

- US Federal CIO Council (http://www.cio.gov)
- UK Government - CIO Council (http://www.cabinetoffice.gov.uk/cio.aspx) Dead link as on 4th Nov 2010, unable to find new official site.
- The Chief Information Officer Concept in E-government: Lessons for Developing Countries. (http://esaconf.un.org/WB/default.asp?action=9&boardid=10&read=3538&fid=97) By D.C. Misra. On United Nations Department of Economic and Social Affairs WebBoard.

Article Sources and Contributors

Enterprise architecture *Source*: http://en.wikipedia.org/w/index.php?oldid=507128596 *Contributors*: 0mark0, 0tsa0, AGK, Alienjim, Andy.k.chang, Antonio Lopez, AntonioVallecillo, Architectchao, Arpabr, Athaenara, Atifmkhan, Bihal, Billaronson, BlackKnight, Boothy443, Brandguru, Brandy Downs, Brian a lee, Can't sleep, clown will eat me, Carl presscott, Catof, Cfoglia, Charles T. Betz, Chris the speller, Clapstick77, Colinwheeler, Colonies Chris, Creacon, Cybercobra, DGG, Daljitbanger, Darp-a-parp, David Waters, David.t.bath, DearMyrah, Deville, Dmccreary, Dr Gangrene, Dubtank, Dvanberg, Dwbrockway, EAIG, EAPRACTICE, Eaguru, Eamteam, Eateacher, EdJohnston, Eddiexx77, Efitu, Eissaf, Emahdizargar, Emrekenci, Enoril, EnterprisePlanner, Erianna, EricStephens, ErikvanB, Erkan Yilmaz, Eustress, Evmako, Factician, Faradayplank, Ff1959, FlyHigh, Foobarnix, Fortmyers, Fred Bradstadt, Gardar Rurak, Gletiecq, Graham Berrisford, H.meligy@ieee.org, Hazarbx, Headbomb, Heppelle, Hervegirod, Hmains, Horatio Huxham, Ian Glossop, IanDBailey, Id027411, In fact, Indon, Inwind, JEH, JForget, Jarble, Jayron32, Jcs2006, Jeff Zhuk, Jmgendron, Jpbowen, Jroschella, Jstehlin, KAtremer, Keithstagner, Khalid hassani, Kjenks, Kpellegrino, Kuru, Kustere, Kvoges, Lakeworks, Larry laptop, Larsonk, Leowashere, Lewisskinner, Lijealso, Lockezachman, Lotje, Lradrama, M4gnum0n, MER-C, Mabdul, Mahanchian, Manishpk, Marcoprez, Mark Renier, Marklane0913, MatthewFordKern, MaxHund, Mdd, Metaframe, Michael Hardy, Michig, Mikeyboysan, Mirkoki, MithrasPriest, Mmalotau, Modify, Moejorris, Momo san, Monash123, Mormegil, Mudgen, Nav102, Nazrani, Ncw0617, Niceguyedc, Nickmalik, Nocontributionsever, Noru66, Ohnoitsjamie, Oicumayberight, Ontist, Per.foldager, Petko, Picus viridis, Pkearney, Poppafuze, Poweroid, Qartman, Qatestdev, R'n'B, Raholijaona, Raju bhupesh, Rapsar, Rfoard, Ricardus64, Rich Farmbrough, Richard R White, Richard.McGuire88, Richard@lbrc.org, RichardVeryard, Roleplayer, Ron Gaba, Ronz, Roypittman, Riverapadilla, Rudytheman, SWA2, Saccland, Saigeethamn, Samhiggins1973, ScottWAmbler, Sct72, Senright, Shajela, Sightreadernow, Signalhead, Somerwind, Sonyasunshine, Spanglej, Sri1968, SunSw0rd, Sutch, Svtveld, Toheller, Tom Corn, Tonyshan, Tribaal, Tschirl, Turnb345, UdayanBanerjee, Vandvik, Veinor, Vicpulido, VishalVPatil, VivekRaman, Winterst, Yahya Abdal-Aziz, Ylvabarker, YoavShapira, Yogishpai, 雁太郎, 389 anonymous edits

Enterprise architect *Source*: http://en.wikipedia.org/w/index.php?oldid=509610622 *Contributors*: 2602:306:CD24:CF80:21C:B3FF:FEB9:F5D3, Ace of Spades, Appraiser, Architectchao, Berny68, Brandy Downs, Bwesche, CBM, Catof, Cobaltbluetony, DGG, Dariopy, EAPRACTICE, Ensa, GardmanVS, Greg Zorne, Happyuk, Headbomb, Hmains, ICloche, Ian Glossop, IceCreamAntisocial, JHunterJ, Jclouse, JorgePeixoto, Kimleonard, King Lopez, Kuru, Lockezachman, M4gnum0n, MBisanz, Marklane0913, Maxis ftw, Mboverload, Mdd, Moringadan, Nickmalik, Nux, OceanKiwi, Peter Campbell, Phronko, Pravin Gupta, Rharner, Rich Farmbrough, RichardVeryard, Rjwilmsi, Roleplayer, Ron Gaba, Ronz, RoySmith, Ruud Koot, Simple Bob, Sn2000, Sonyasunshine, Svtveld, The Thing That Should Not Be, Yogishpai, 70 anonymous edits

Enterprise Architecture Assessment Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=479397322 *Contributors*: Brian a lee, Cat4567nip, DGG, Headbomb, Jpbowen, MatthewVanitas, Mdd, 2 anonymous edits

Enterprise architecture planning *Source*: http://en.wikipedia.org/w/index.php?oldid=440072947 *Contributors*: Denisarona, Headbomb, JForget, Kjkolb, Mdd, Mirkoki, Nav102, Vprajkumar, Woohookitty, 3 anonymous edits

Enterprise Architecture Management *Source*: http://en.wikipedia.org/w/index.php?oldid=494950277 *Contributors*: Brandguru, Chowbok, ClaretAsh, DianaNier, Dinishare, Edward, Ehheh, Headbomb, Kuru, Manishpk, Mdd, MrOllie, R'n'B, Redrose64, RichardVeryard, Saccland, Smid12, Van der Hoorn, Yogishpai, 6 anonymous edits

Enterprise Architecture framework *Source*: http://en.wikipedia.org/w/index.php?oldid=469576512 *Contributors*: AntonioVallecillo, Appraiser, Architectchao, Boppet, ChanochWiggers, Coherers, Colinwheeler, Conrad Hughes, David.t.bath, Dcannin, Drilnoth, Drstrangeluv25, Dynamicedge, Erkan Yilmaz, Gaius Cornelius, Graham Berrisford, Griffin700, Headbomb, Hervegirod, IanDBailey, Iohannes Animosus, Isometric, J.delanoy, JForget, JohnCD, Kevinleesmith, Krohorl, Leowashere, M1ss1ontomars2k4, Marc Schade, Mdd, Miljoshi, Nav102, Nick Number, Nickmalik, Pinaghosh, R'n'B, RfHilliard, RichardVeryard, Roebuckr, Ronz, Rriverapadilla, S650588, SWA2, Samhiggins1973, SchreiberBike, SunSw0rd, Tabletop, Tony1, UnitedStatesian, Wikitect, Woohookitty, 雁太郎, 71 anonymous edits

Architecture Patterns (EA Reference Architecture) *Source*: http://en.wikipedia.org/w/index.php?oldid=402836069 *Contributors*: 4th-otaku, AlastairIrvine, Archknight, Avicennasis, BD2412, Beland, BenjaminHeitmann, Bobo192, CamilleLM, Christian Storm, Christian75, Codeczero, Davidfstr, Ephillipe, Honette, Jafeluv, Janggeom, Jjalexand, Jpbowen, Katjah, Kenyon, Kku, LtPowers, M4gnum0n, Madmikey, Martarius, MartinSpamer, Nav102, Naveen607, Pnm, Ptrb, Rory O'Kane, Rpawson, Rursus, Rwwww, Sae1962, Sandeep ubs, Sbywater, Scientific29, Spdegabrielle, Susanpadura, Thedjatclubrock, Tomrbj, Topbanana, Vinsci, Wlievens, 48 anonymous edits

Enterprise Architecture framework *Source*: http://en.wikipedia.org/w/index.php?oldid=469576512 *Contributors*: AntonioVallecillo, Appraiser, Architectchao, Boppet, ChanochWiggers, Coherers, Colinwheeler, Conrad Hughes, David.t.bath, Dcannin, Drilnoth, Drstrangeluv25, Dynamicedge, Erkan Yilmaz, Gaius Cornelius, Graham Berrisford, Griffin700, Headbomb, Hervegirod, IanDBailey, Iohannes Animosus, Isometric, J.delanoy, JForget, JohnCD, Kevinleesmith, Krohorl, Leowashere, M1ss1ontomars2k4, Marc Schade, Mdd, Miljoshi, Nav102, Nick Number, Nickmalik, Pinaghosh, R'n'B, RfHilliard, RichardVeryard, Roebuckr, Ronz, Rriverapadilla, S650588, SWA2, Samhiggins1973, SchreiberBike, SunSw0rd, Tabletop, Tony1, UnitedStatesian, Wikitect, Woohookitty, 雁太郎, 71 anonymous edits

Enterprise Architecture Body of Knowledge *Source*: http://en.wikipedia.org/w/index.php?oldid=444609380 *Contributors*: David.T.Bath, David.t.bath, Greyskinnedboy, Jackfork, Jpbowen, Nickmalik, Quarilari23, RichardVeryard, Robomanx, Ronz, Tabletop, 雁太郎, 10 anonymous edits

Generalised Enterprise Reference Architecture and Methodology *Source*: http://en.wikipedia.org/w/index.php?oldid=484018475 *Contributors*: Cat4567nip, Estienne, J04n, KTachyon, Mdd, R'n'B, WikHead, 2 anonymous edits

IDEAS Group *Source*: http://en.wikipedia.org/w/index.php?oldid=476064833 *Contributors*: Aecis, AvicAWB, Folajimi, IanDBailey, Leggattst, Mdd, Merlion444, Richard R White, The Anome, 9 anonymous edits

RM-ODP *Source*: http://en.wikipedia.org/w/index.php?oldid=502256247 *Contributors*: AntonioVallecillo, Cat4567nip, EdMcMahon, Husond, Kaluzman, Mdd, Miym, Nasa-verve, Nickg, Odalet, RichardVeryard, The s0rc3r, 14 anonymous edits

The Open Group Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=508057779 *Contributors*: Apokrif, BatesyM, Cat4567nip, Chowbok, CiTrusD, Colinwheeler, Craigrmartin, Damian Gawlowski, Decoy, Desphi, Discospinster, Dominik Kuropka, Dwelliott, Edhgoose, Elpecek, ErikvanB, Faramir4, FireballDWF2, Georgewilliamherbert, Ghettoblaster, Gmjohnston, Graham Berrisford, GrahamHardy, Greyskinnedboy, Hassan210360, Hervegirod, Hojimachong, IP 84.5, IanDBailey, IvanLanin, JForget, Jamelan, Jeff Zhuk, John Vandenberg, Jonste, Juliank, Kmorozov, Laoyao99, Larsonk, Lokpest, Mahanchian, Maria C Mosak, Markphahn, Mdd, Mgl795, Mohamad Hani ELJAMAL, Nurg, Phantom309, Pittspilot, Prof 7, QuiteUnusual, RHaworth, Richard R White, RichardVeryard, Ronz, Rvodden, Sgryphon, Spike Wilbury, Stephen, Stuartyeates, SunSw0rd, Tdubendorf, TechnoMorphWiki, Timdwilliams, ToolPioneer, Turnb345, Uppalj, Yan Kuligin, Zanaq, 138 anonymous edits

Integrated Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=467541990 *Contributors*: Cat4567nip, Iznogoud, Mdd, Mtaksrud, Ontist, R'n'B, Rpearlman, Skysmith, T@nn, 4 anonymous edits

CLEAR Framework for Enterprise Architecture *Source*: http://en.wikipedia.org/w/index.php?oldid=467370505 *Contributors*: Cat4567nip, Mdd, Nn123645, Ricardus64, Ronz, Scriberius, 4 anonymous edits

OBASHI *Source*: http://en.wikipedia.org/w/index.php?oldid=500752374 *Contributors*: Adi4094, Cat4567nip, Chowbok, Chris the speller, Dadadi, DanielV UK, Fergus Cloughley, Gary King, Jason Quinn, John of Reading, Kuru, LilHelpa, Lotje, Mdd, Rjjm, Rjwilmsi, 31 anonymous edits

Information Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=503297411 *Contributors*: Adrianrgcampbell, Cat4567nip, Giraffedata, Infotect, Malcolma, Mdd, Raja Angamuthu, The squaw on the hippopotamus, 21 anonymous edits

Zachman Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=508925218 *Contributors*: Aarktica, Adnanmukhtar, AngelovdS, AnneBoleyn1536, Blaxthos, Boppet, BryanG, CQ, CasperGoodwood, Cat4567nip, Chowbok, ChristopheS, CommonsDelinker, DEddy, DGG, Darklilac, Deville, Duke Ganote, EagleFan, EdJohnston, Elpecek, Fang 23, Fredrick day, Gaius Cornelius, Gark08, GoingBatty, GoodStuff, GrWikiMan, Graham Berrisford, Gregbard, Greyskinnedboy, Grgarza, Gzornenplatz, HGB, Hagge, Hu12, Ian Pitchford, Ideasintegration, J04n, John of Reading, Jpzachman, Kmorozov, Larrygoldberg, Larsonbennett, Len Morrow, LilHelpa, Lockezachman, Lousyd, Louv, Mahanchian, Mandarax, Maria C Mosak, Mdd, Metaframe, Metaman1, Mhavila, Mlibby, Muhandes, Natalya, Next2u, Nickmalik, Ontist, Paddles, PaulTanenbaum, Phogg2, Phwedh, R'n'B, Rcbapb, RichardVeryard, Ron Gaba, Ronz, Sbowers3, ScottWAmbler, Sct72, Skapur, SunSw0rd, Tcrowley7, Tennekis, Thehelpfulone, Tom Corn, Woohookitty, Ynhockey, 雁太郎, 128 anonymous edits

Department of Defense Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=508840465 *Contributors*: 5994995, 5cnts, Aldryd, AlephGamma, Athaenara, Atpco, Brons, Buckshot06, Bunnyhop11, Cask05, Cat4567nip, Deemery, Doug Bell, ESkog, Galoubet, Garion96, Gletiecq, Harryt99, Hervegirod, Hu12, IanDBailey, Igfrace, Izno, Jaberwocky6669, Jean-Philippe LERAT, John.szucs, Jstehlin, Juniorj1967, Kitdaddio, Leggattst, Louv, MME, MRqtH2, Maria C Mosak, Mark Arsten, Mark Renier, Mdd, Mechthold54, Metaframe, Mjchonoles, Mwilmshurst, Mygerardromance, Nick Number, PaulHanson, PeterGerstbach, Pglehmann, Pshoman, Rg348, Richard R White, RichardVeryard, Rikenmyer, Robertbowerman, Ronz, SWA2,

Sauzuk, Senright, Skysmith, Starsman, Sun Creator, Tabletop, TechBuzz28, TheKMan, Thomas Willerich, Tremnai55, UpdateMiller, Wikisteff, 雁太郎, 109 anonymous edits

MODAF *Source*: http://en.wikipedia.org/w/index.php?oldid=501219337 *Contributors*: Adam-griffiths, Adeio, Bearcat, Bigglescat, Branedancearj, CIO-MODAF, Carlbarck-holst, Cat4567nip, EoGuy, Esprit15d, Hammersoft, Helen Peacop, Hervegirod, Huku-chan, IanDBailey, Isopropyl, JHunterJ, Josee Bourdage, Kwekubo, Leggattst, Maria C Mosak, Mark Renier, Mdd, Mjchonoles, MrOllie, Pearle, PeterGerstbach, Porqin, Rayc, RichardVeryard, Rjwilmsi, Robertbowerman, Roypittman, Visvais, WOSlinker, Welsh, Wikitect, WolfgangFahl, 39 anonymous edits

NATO Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=467367959 *Contributors*: Cat4567nip, Fabrictramp, GregorB, IanDBailey, Marc Schade, Mdd, Sarahnortonviolin, Wikitect, WolfgangFahl, 2 anonymous edits

AGATE Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=482992225 *Contributors*: Bobblehead, Cat4567nip, Hervegirod, Mais oui!, Maria C Mosak, Mdd, Premeditated Chaos, Rich Farmbrough, RichardVeryard, Switchercat, The Evil Spartan, 1 anonymous edits

Government Enterprise Architecture *Source*: http://en.wikipedia.org/w/index.php?oldid=453799351 *Contributors*: Bkonrad, Jpbowen, LilHelpa, Samhiggins1973, Seattlenow, Shiftchange, Tony1, VanessaDS, 2 anonymous edits

FDIC Enterprise Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=467367616 *Contributors*: Cat4567nip, Chowbok, Edward, Gregg 84102, Ground Zero, Mdd, Psantora, Will20036, 2 anonymous edits

Federal Enterprise Architecture *Source*: http://en.wikipedia.org/w/index.php?oldid=465659030 *Contributors*: Adi4094, Brandguru, Cat4567nip, Chowbok, Cmholm, Crkey, Daveshawley, David.t.bath, Dmccreary, DoctorW, Epolk, Giraffedata, Gletiecq, GoodStuff, Ihuxley, Ironwolf, Jpbowen, JubalHarshaw, Kalai545, Kitdaddio, Krauss, Kuru, Lanma726, Lantrix, Lockezachman, Louv, Maria C Mosak, Mark Arsten, Mark Renier, Mdd, Nageh, Nisteditor2, PaulHanson, Piels, Pinethicket, RichardVeryard, Ronz, Senright, Sheardsheard, Snidhi22, The Horst Mann, Tony1, ToolPioneer, Traveler23380, 雁太郎, 36 anonymous edits

NIST Enterprise Architecture Model *Source*: http://en.wikipedia.org/w/index.php?oldid=467367886 *Contributors*: Apoc2400, Cat4567nip, MartinZwirlein, Mdd, Neutrality, 2 anonymous edits

Treasury Enterprise Architecture Framework *Source*: http://en.wikipedia.org/w/index.php?oldid=467368924 *Contributors*: Cat4567nip, Euchiasmus, Mdd, Rossj81, Saeed.Darya, 2 anonymous edits

Enterprise life cycle *Source*: http://en.wikipedia.org/w/index.php?oldid=502768974 *Contributors*: Jean.julius, Kjkolb, Mdd, Mr pand, Pm master, Scmbwis, Woohookitty, 11 anonymous edits

ISO 12207 *Source*: http://en.wikipedia.org/w/index.php?oldid=457762836 *Contributors*: Alainr345, AlephGamma, Alopex0, BenAveling, Dekimasu, Fieldday-sunday, Herbee, Italinux, Jamelan, Khalid hassani, Limbo socrates, Lmatt, M.e, Mavetx, Mdd, Mitch Ames, Mxn, Nick Number, Ninly, Pearle, Phil Boswell, Pichpich, Radagast83, Siliconshells, Taibah U, Thebestofall007, Tim1357, UnitedStatesian, Vikebo, 33 anonymous edits

Systems Development Life Cycle *Source*: http://en.wikipedia.org/w/index.php?oldid=458627523 *Contributors*: 90, ABF, AGK, AbbasSarfraz, Adamgibb, Ahoerstemeier, Aitias, Ajbrulez, Albanaco, AlephGamma, Allan McInnes, Allens, Allmightyduck, Allstarecho, Alpha Quadrant (alt), Andaryj, Anonymous Dissident, Ansell, Arjun01, Arunsingh16, Avaloan, BasilBoom, Beiber the justin, Bfigura's puppy, Bilby, BillyPreset, Bobo192, BrianY, CFMWiki1, CalebNoble, Can357, CanadianPenguin, Cbdorsett, Chilllls, Chris 73, Chrislk02, Chsh, Cliffydcw, Closedmouth, Cochiloco, Copyeditor22, Cory Donnelly, Croepha, DARTH SIDIOUS 2, DEddy, DGJM, Dandv, Danielphin, David.alex.lamb, DavidDouthitt, DeadEyeArrow, Devarishi, Dgw, Djsasso, DrKranium, DuncanHill, ESkog, Eastlaw, Ebessman, Ekren, Enauspeaker, Ensign beedrill, Epbr123, Erkan Yilmaz, Evaders99, FalconL, FiRe, Flyerwolf, GD 6041, Gaff, Gamcall, GarbagEcol, Gert7, Gilliam, Gioto, Graeme Bartlett, Grafen, Happysailor, Harsha.nagaraju, Hghyux, Holsen266, Hughiki, Hutcher, Hydrogen Iodide, HyperSonic X, Informatwr, IvanLanin, J.delanoy, J04n, JaGa, JamesBWatson, Jdg12345, Jeff G., Jeremy Visser, Jim1138, Jinian, Jncraton, Joe4210, John Vandenberg, John.stark, Johnjeric, Jonverve, Josephedouard, Jpbowen, Kalai545, Kdridder, Kenyon, Kesla, Kevin, Khalid hassani, King of Hearts, Kingpin13, Kku, KnowledgeOfSelf, Kr suraesh, Kuru, Kurumban, Kushal one, L Kensington, LAX, Latka, Lectonar, Liftarn, Little Mountain 5, Lousyd, Lugia2453, Luna Santin, M4gnum0n, Madman2001, Maghnus, Marek69, Mark Renier, Materialscientist, Matt.sigs, Mdd, Meisterflexer, Michael Hardy, Mick1990, Mike Rosoft, Mikesc86, MrOllie, Mrwojo, Mummy34, NTox, Natkeeran, Natl1, Neurophyre, Niaz, Nkattari, NuclearWarfare, OMouse, Ocaasi, Octahedron80, Ohnoitsjamie, Oliverclews, Omicronpersei8, Orange Suede Sofa, PS., Pascal666, Paxsimius, Philip Trueman, Piano non troppo, Pikajedi3, Pinethicket, Pingveno, Pip2andahalf, Pluke, Pnm, Qaiassist, Qrsdogg, R'n'B, RadioFan, Ranjithsutari, Raymondwinn, Recognizance, Rich Farmbrough, Rjwilmsi, Ronz, Rooseveltavi, Rwwww, Salvio giuliano, Sam Hobbs, Sarloth, Satishshah123, SchmuckyTheCat, ScottWAmbler, Seahorseruler, SentientSystem, Seth Ilys, Shadowjams, ShwetaCool, Sigma 7, SigmaJargon, Specs112, Srushe, Steelchain, SteveChervitzTrutane, Stickee, Strawny02, Suneelkushwaha, Supran, Swep01, Syrthiss, Tanvir Ahmmed, Tedder, Tee Owe, Teles, Teohhanhui, The Anome, ThePointblank, TheProject, TheTallOne, ThomasO1989, Thomasjeffreyandersontwin, Throwaway85, Tide rolls, Tommy2010, Tony1, Tonyshan, Transcendence, TwilligToves, Vashtihorvat, VectorThorn, Versus22, Verypprettyfish, WikHead, Wiki alf, Wikipelli, Wimt, Woohookitty, Wspencer11, Wtmitchell, Xphile2868, Xsmith, Yeng-Wang-Yeh, Yogi de, Yossiea, Zondor, 1052 anonymous edits

Technology Life Cycle *Source*: http://en.wikipedia.org/w/index.php?oldid=469659385 *Contributors*: Auntof6, Bachrach44, DragonHawk, Giraffedata, Hbent, Hermitage17, Ipsofacto, Iridescent, LittleWink, Markhurd, McGeddon, Mdd, Nick Number, Nopetro, Phantomsteve, R'n'B, Rjwilmsi, Swferrier, Tony1, Woohookitty, Yworo, 8 anonymous edits

Whole-life cost *Source*: http://en.wikipedia.org/w/index.php?oldid=507604456 *Contributors*: Gnfnrf, Helenalex, Hmains, JonathanBentz, Kku, LilHelpa, Martarius, Mdd, Motorworld, MrOllie, Mrwojo, Mîthrandir, Nick Number, OceanKiwi, Paul W, ProjMngt, Quarl, Robtrob, Scmbwis, Smobbl Bobbl, Velella, Woohookitty, 10 anonymous edits

Enterprise modelling *Source*: http://en.wikipedia.org/w/index.php?oldid=489184814 *Contributors*: AndrewHowse, Banana Concoction, Chris the speller, Cnbrb, EagleFan, EnterpriseModeller, Equilibrioception, EwokiWiki, Giraffedata, GrWikiMan, Hazmat2, J04n, Jncraton, Keithh, KvanHaperen, LilHelpa, Mdd, Noble Krypton, PongoPrint, Purge me, R'n'B, Razorbliss, ReyBrujo, RichardVeryard, ScottWAmbler, Some standardized rigour, Stanio, Telecomtom, TheTito, Wbm1058, Woohookitty, 雁太郎, 27 anonymous edits

Business analyst *Source*: http://en.wikipedia.org/w/index.php?oldid=509691661 *Contributors*: 02design, 2KT, 3ta, R, 9ign, AbbasSarfraz, Albattar, Anagha dhumal, Andrew Kay, Avicennasis, Azcontributor, Baanalyst, Baspecialist, Benglish11, Bmitchelf, Boing! said Zebedee, Brick Thrower, Chelseawoman1, Choalbaton, Closedmouth, Cnbrb, Cobaltcigs, Cpuwhiz11, DVdm, Daugustine, Dave601, DavidDouthitt, Davidovic, Davidvilson2009, Deereav, Dfarnham, Discospinster, Dr.frog, Dwandelt, ESkog, Edwardmking, Elainarufs, Evans1982, Freek Verkerk, Frequencydip, Fusionmix, GDonato, Gadly, Gary King, Geeteshjain, Ghaag, Goethean, Greyskinnedboy, Gromlakh, Gurch, Harro5, Heirpixel, Hitrish, Humblefool, I dream of horses, Indexed card, Irmtraining, J.delanoy, Jackofwiki, Jasper432, Jaychowrocksmysocks, Jcampsall, Jeh, Jjalexand, JonathanWeaver, Josbald1975, JulianSammy, Justaba500, Kalpanatikoo, Kateshortforbob, Kathode raytube11, Kbdank71, Kevin Brennan, Kgrr, Khalid hassani, Kinkyfly, Kpalion, Kuru, Kx1186, L Kensington, Lightlowemon, LilHelpa, M4gnum0n, Mactin, Manytexts, Maria C Mosak, MarkGallagher, Mathewsfish, Matthew0028, Mdd, Michael Hardy, Mindmatrix, Mion, Monymirza, Movses, MrOllie, NawlinWiki, Nick Number, Notinasnaid, NuclearWarfare, Odie5533, Officiallyover, Ohnoitsjamie, Oli2007, Outriggr, Pendse pradeep, Petergjansen, QuiteUnusual, R'n'B, Radagast83, Randhirreddy, Rklawton, RobinGoldsmith, Rodney Boyd, Ronz, SCEhardt, Santosh kumar g, Saxifrage, Scratz, Siegfried1, Slon02, Stuartlachlan, Stupid Corn, Sunshinedaydream95, Taliped, Tassedethe, Terzett, The Land, The Rambling Man, The-sloans, TheDJ, Thingg, Thumbarger, Tide rolls, Vdavisson, Vwtinfo, Wacko39, Wenat, Wesmith4, Wikicsb, Zzuuzz, 432 anonymous edits

Systems analysis *Source*: http://en.wikipedia.org/w/index.php?oldid=509706422 *Contributors*: 1ForTheMoney, Aitias, Al.Ragsdale, Albamuth, AllTheCoolNamesAreAlreadyTaken, Architectchao, Ask123, Beowulph, BigHaz, Bigs slb, Bobbyanalog, Bobo192, Brian48, Btyner, Burzmali, CMG, Cask05, Charleenmerced, Cyrus Grisham, DVdm, Deipnosophista, EagleFan, El C, Erkan Yilmaz, Fooziex, Freakmighty, Goethean, HusiGeza, Ian Tabi, Impaciente, Irmtraining, Jac16888, Jeffmartincg, Jesgab, Jheald, Johnbrownsbody, JonHarder, Jpbowen, Kenyon, Kingturtle, Kuru, Larsobrien, Liridon, Logicalgregory, Luna Santin, M Preston Sparks, Malcolmxl5, Maria C Mosak, Mark Renier, Materialscientist, Maurreen, Mblumber, Mcsharry, Mdd, Mnent, Mr Stephen, MrOllie, Mydogategodshat, Mårten Berglund, NawlinWiki, Nbarth, Nick Green, Nono64, Octahedron80, Philip Trueman, Philosopher06, Piano non troppo, Pinethicket, Qst, R'n'B, RJBurkhart, Rajah, RattleMan, Remuel, Riana, Rich Farmbrough, Ronz, S0aasdf2sf, Shreshth91, SimonClinch, Skittleys, Soresumakashi, Srikant.sharma, Stevertigo, Straker, Strangnet, Stypex, Sverdrup, Tawker, Tbackstr, Ted Wilkes, The Missing Hour, The Rambling Man, Tide rolls, Toxickore, Velella, WereSpielChequers, Wiki Roxor, Willking1979, Wpociengel, X!, Yamamoto Ichiro, Zoara, 222 anonymous edits

Information architecture *Source*: http://en.wikipedia.org/w/index.php?oldid=509549466 *Contributors*: 2602:306:CD24:CF80:21C:B3FF:FEB9:F5D3, A3RO, Aapo Laitinen, Aaronbrick, Abialek, Ajcheng, Angela, Asist, Atomiq, Bevcorwin, Bill.albing, BirgerH, Brad7777, Brick Thrower, Burkhard, Capricorn42, Captmondo, Chrisfurniss, Clayoquot, Craigb81, Creacon, DEddy, DanimovIA, Danimovski, DerHexer, DigitalWikiMan, Dobrien, Eddiexx77, Edward, Ehheh, Ekillaby, Emorrogh, Enric Naval, Erliptac, Fenice, Flyguy649, Frequencydip, GlassFET, Graham Berrisford, Greenmtncville, Harvardnet, Havanafreestone, Hbent, Hobartimus, Horatio Huxham, Icseaturtles, Infophiliac, IvanLanin, J04n, Jakew, JimmycurN, Jmelody, Jomackiewicz, Jpbowen, Justin2007, KPH2293, Kaganer, Kauczuk, Kellylautt, KeyStroke, Koavf, Kostmo, LLarson, LeeHunter, LilHelpa, LockeShocke, Lostnumber747, Lotje, Luís Felipe Braga, Lycurgus, Mahanchian, Mandolinface, Marcok, Martarius, Marty computerscientist, Masao, Mbria, Michael Hardy, MrOllie, Mymarpie, NicHB7, Nickmalik, Nighthawk2050, Novacough, Oconar, Of.information.architecture, Ohnoitsjamie, Oicumayberight, Onepd, Orangewarp, Outriggr, Parham, Pavel Vozenilek, Pboersma, Pema, PhilipR, Phoebe, Pinethicket, Prof 7, Pucciar, Quarty, Resmini, RichMorin, Ronz, S.K., Salvadorious, Scetoaux, SeanGustafson, Seth.lilly, SilvrWings, Simon.raistrick, Skeejay, Smallman12q, Smoken Flames, Socrates32, Spalding, Speedoflight, Studip101, Tankgrrl2000, TeunSpaans, The Thing That Should Not Be, Timber92, Timo Kouwenhoven, Tobias Bergemann, UXjam, Urhixidur, Veinor, Velvetsmog, Willking1979, Yworo, シ, 277 anonymous edits

Solutions Architect *Source*: http://en.wikipedia.org/w/index.php?oldid=500834296 *Contributors*: 2602:306:CD24:CF80:21C:B3FF:FEB9:F5D3, Aaadonai, Enviroboy, Fyrftd, Glacialfox, Graham Berrisford, Holsen266, Icairns, Jamesont, Jonathanchaitow, Kuru, Mdd, Michael.W.Meissner, Paulhussein, RedWolf, Ronz, 43 anonymous edits

Software architect *Source*: http://en.wikipedia.org/w/index.php?oldid=509549644 *Contributors*: 2602:306:CD24:CF80:21C:B3FF:FEB9:F5D3, Aladdin Sane, Allan McInnes, Alucard (Dr.), Andreas Kaufmann, Antandrus, Anwar saadat, Bhny, Bigbluefish, Boson, Bovineone, BrianFennell, Cybercobra, DabMachine, Daewoochong, Deville, Doryexmachina, Doug Bell, Drable, Drheaton, DukeyToo, Erianna, EricStephens, Ervinn, Gail, Hetar, HiEv, Hydrox, Jatkins, Java4food, JimVC3, JonHarder, K.Nevelsteen, Koffie, Kompere, Kuru, Leoluz, Longshot14, Maria C Mosak, Max Terry, Mdd, MichaelBillington, MikelZap, Mr4tastic, Mütze, Neuralwarp, Nickmalik, Nightscream, Normxxx, Pinecar, Poosari, RichardAlexanderGreen, Robth, Ronz, RoyBoy, Ryan Roos, SamJohnston, Sapphic, Swampdaddy87, Techmage, Tlatoani, UnitedStatesian, Vipinhari, Wigren, Wikilobbyist, Willpeavy, Wspencer11, XLerate, Xtremeways, 82 anonymous edits

Systems architect *Source*: http://en.wikipedia.org/w/index.php?oldid=509549627 *Contributors*: 2602:306:CD24:CF80:21C:B3FF:FEB9:F5D3, 9Nak, Ablqadir, Allan McInnes, Andreas Kaufmann, Bigbluefish, Bobblehead, BrainyBabe, CLW, CardinalDan, Chris Pickett, D6, DabMachine, Deville, DonFiresmith, Drieux, Drmies, EagleFan, Furrykef, Gilliam, Hetar, Iridescent, J04n, JoanZehrt, K.Nevelsteen, Kmorozov, Longshot14, Louv, Luís Felipe Braga, Maria C Mosak, Markhu, Mdd, MichaelBillington, Mrgates, Normxxx, Open2universe, Philip Sutton, Psychohistorian, RichardVeryard, Robth, Ronz, Ryan Roos, SEI Publications, Sapphic, SchreiberBike, Sct72, Sempf, Soap, Texclayton, Toxickore, Wbm1058, WikHead, Woohookitty, Yousou, Zac Gochenour, 61 anonymous edits

Project manager *Source*: http://en.wikipedia.org/w/index.php?oldid=509742958 *Contributors*: Aff123a, Akalanaki, Alexolson, Antandrus, Artemis Fowl Rules, Axelberube, Banej, Bencherlite, Bhalchandratk, Blair Sutton, BrainyBabe, C45207, Chiefmanzzz, Chowbok, Chris Roy, Cleduc, Cloud10pm, Co1Spain, Cymru.lass, Dandriani, David Biddulph, Deville, Digitric, Dineshasewwandi, Discospinster, Dlohcierekim, Dogears, Eeekster, Ehheh, Enviroboy, Evans1982, Ferdilo, Fernandocastor, Ff1959, Ft1, Glane23, Globalprofessor, Gobonobo, Graeme Bartlett, Haikon, Heron, Hgfernan, Hoof Hearted, Hubertus, Huffry94, IMsupersmith, Iridescent, Jacobsharps, Jagra, Jazzy Diva, Jcampsall, JensenDied, Jsiraj, KGun10, Karmona, Keith Biglin, Keith Richards UK, KimBecker, Kku, LeonMichaelSmith, Lordbrocket, Luna Santin, MQuinn, Mdd, Mentifisto, Mfo321, Michelle2mmb, Mkoval, MrOllie, Mtcedar, Muazen, Nan-bread666, Nankivel, Nazmanager, Od Mishehu, Oicumayberight, Ojophoyimbo, OlEnglish, Piano non troppo, PingPongBoy, Pjganon, Ploum's, Pm master, QuiteUnusual, R'n'B, RA0808, RJFJR, Richi, Richieman, Rillian, Rundquist, S.K., Saber girl08, Sean Whitaker, Shanes, Sisalto, Stevenmitchell, Synique, The Thing That Should Not Be, Thebluemanager, Travis99, Vanmarcus, Vavilen, Vincehk, Voldemortuet, Wdyoung, Wik, Wiktoria, Wndola, Wtmitchell, Zigger, 195 anonymous edits

Project management office *Source*: http://en.wikipedia.org/w/index.php?oldid=508831947 *Contributors*: Akbradford, Asamawal, Ashrafhashem, BirgitteSB, Boudewijn, Cespa, CharlotteWebb, Chasep2010, Chris0427, DanReedPMP, Darrentcook, DavidAndres, DemolitionMan, DerHexer, DickieTruncheon, Donalmc, Drullewatz, EPRNYC, Eduardo6028, Flashart1, Fortdj33, Greyskinnedboy, J.delanoy, Jwestland, Karateguyindc, KathrynLybarger, Kjkolb, Kku, Kokilads, Kuru, Lightfair, Lindsayascott, Mapador, Mattg82, MrOllie, NickelShoe, Paraparam Bigot, Pascal666, Pearle, Pm master, Qaiassist, Rodii, SDC, SatuSuro, Scmbwis, SmartGuy Old, TastyPoutine, The Epopt, TheDJ, U+003F, Vanished User 1004, Xp54321, YUiCiUS, 70 anonymous edits

Chief information officer *Source*: http://en.wikipedia.org/w/index.php?oldid=498379643 *Contributors*: 72Dino, Aecis, Aldie, Alphachimp, Andrewmin, Aranea Mortem, Benjamin Mako Hill, BlueSkyNoRain, Bruce404, Buster7, C172rg, CIOGuru, Cebess, ChristianBk, Comfychaos, Curtis Clark, Cybercobra, Dandv, Daniel73480, David Shay, Davidp, Dismas, Diza, Djharrity, Dmccreary, Dub617, Dukeofgaming, EFormsBlogger, Egil, Ehheh, Fbarw, Gadfium, Gaia Octavia Agrippa, Gatemansgc, Greyengine5, Groverxv, Hede2000, Hqb, Ian Pitchford, Iswive, Iznogoud, JForget, Jamalystic, Jaysee21, Jazmatician, Jmilestaylor, Johnloia2112, JonHarder, Joseph.strunz, Jplarkin, Julsssark, Jvlock, Kuru, Laran.evans, Leslie Mateus, LiDaobing, Lotje, Mairi, Millstream3, Mitchello, Mote, Mpeisenbr, MrOllie, Mrflip, Neutrality, Nslonim, Nurtsio, Ohnoitsjamie, OminousSkitter, OnceAlpha, OneGyT, Pan Dan, PaulHanson, Poweroid, Pravin.takseveryday, Rhobite, Rich Farmbrough, Rjwilmsi, Robert Merkel, Robofish, Rock2e, Ronz, Rrastin, Schneelocke, Sebesta, Shajela, Smpclient, SusanLesch, Svetovid, Swineinfluenza, Tellmewhy1, Tempodivalse, The Thing That Should Not Be, Thecheesykid, Themfromspace, Thiagoeh, Thomasjl, Vague, Wiki3 1415, Woohookitty, Yarnalgo, Zelig123456, Zondor, 148 ,דוד ש anonymous edits

Image Sources, Licenses and Contributors

File:Architectural Levels and Attributes.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Architectural_Levels_and_Attributes.jpg *License*: Public Domain *Contributors*: Federal Enterprise Architecture Program Management Office, OMB

File:EAAF Maturity levels.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:EAAF_Maturity_levels.jpg *License*: Public Domain *Contributors*: US OBM

File:Information and IT-Enabled Performance Improvement Lifecycle.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Information_and_IT-Enabled_Performance_Improvement_Lifecycle.jpg *License*: Public Domain *Contributors*: US OBM

File:EAAF assessment table.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:EAAF_assessment_table.jpg *License*: Public Domain *Contributors*: US OMB

Image:Levels of Enterprise Architecture Planning.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Levels_of_Enterprise_Architecture_Planning.jpg *License*: Public Domain *Contributors*: Federal Aviation Administration

File:EAMMF Version 1.1 - Maturity Stages, Critical Success Attributes, and Core Elements.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:EAMMF_Version_1.1_-_Maturity_Stages,_Critical_Success_Attributes,_and_Core_Elements.jpg *License*: Public Domain *Contributors*: Mdd

File:NIST Enterprise Architecture Model.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:NIST_Enterprise_Architecture_Model.jpg *License*: Public Domain *Contributors*: The Chief Information Officers Council

File:Evolution of Enterprise Architecture Frameworks.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Evolution_of_Enterprise_Architecture_Frameworks.jpg *License*: Public Domain *Contributors*: Original by Stephen Marley, NASA /SCI, update by Marcel Douwe Dekker

Image:Architecture framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Architecture_framework.jpg *License*: Public Domain *Contributors*: Rob Thomas, Director and Phil Cullen, Policy Analyst, Technology and Architecture Group, Office of Information and Technology

Image:FEA Reference Models.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:FEA_Reference_Models.jpg *License*: Public Domain *Contributors*: CIO Councel

Image:Layers of the Enterprise Architecture.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Layers_of_the_Enterprise_Architecture.jpg *License*: Public Domain *Contributors*: Niles E Hewlett, PMP CEA, Enterprise Architecture Team, USDA-OCIO

File:Enterprise Architecture Domain Reference Architecture.JPG *Source*: http://en.wikipedia.org/w/index.php?title=File:Enterprise_Architecture_Domain_Reference_Architecture.JPG *License*: Public Domain *Contributors*: Naveen607

File:4+1 Architectural View Model.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:4+1_Architectural_View_Model.jpg *License*: Creative Commons Attribution-Sharealike 3.0 *Contributors*: Marcel Douwe Dekker

File:GERAM Framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:GERAM_Framework.jpg *License*: Public Domain *Contributors*: JG Nell, NIST (redrawn by Marcel Douwe Dekker)

File:GERA Life-Cycle Concept.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:GERA_Life-Cycle_Concept.jpg *License*: Public Domain *Contributors*: JG Nell, NIST (redrawn by marcel Douwe Dekker

File:GERA Enterprise-Entity Concept.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:GERA_Enterprise-Entity_Concept.jpg *License*: Public Domain *Contributors*: JG Nell, NIST (redrawn by Marcel Douwe Dekker)

File:GERA Enterprise-Entity Concept--Type 3.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:GERA_Enterprise-Entity_Concept--Type_3.jpg *License*: Public Domain *Contributors*: JG Nell, NIST (redrawn by Marcel Douwe Dekker

File:GERA Generic-Reference-Architecture Concept.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:GERA_Generic-Reference-Architecture_Concept.jpg *License*: Public Domain *Contributors*: JG Nell, NIST (redrawn by Marcel Douwe Dekker)

File:Enterprise Engineering and the Life-Cycle Concept.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Enterprise_Engineering_and_the_Life-Cycle_Concept.jpg *License*: Public Domain *Contributors*: JG Nell, NIST (redrawn by Marcel Douwe Dekker

Image:PD-icon.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:PD-icon.svg *License*: Public Domain *Contributors*: Alex.muller, Anomie, Anonymous Dissident, CBM, MBisanz, PBS, Quadell, Rocket000, Strangerer, Timotheus Canens, 1 anonymous edits

File:RM-ODP viewpoints.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:RM-ODP_viewpoints.jpg *License*: Creative Commons Attribution-Sharealike 3.0 *Contributors*: Marcel Douwe Dekker

Image:TOGAF ADM.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:TOGAF_ADM.jpg *License*: Public Domain *Contributors*: Stephen Marley, NASA /SCI

File:DoD Standards-Based Architecture Planning Process.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoD_Standards-Based_Architecture_Planning_Process.jpg *License*: Public Domain *Contributors*: Department of Defense

Image:CLEAR RELATIONS NG-1-.PNG *Source*: http://en.wikipedia.org/w/index.php?title=File:CLEAR_RELATIONS_NG-1-.PNG *License*: Creative Commons Attribution-Sharealike 3.0,2.5,2.0,1.0 *Contributors*: Ricardus64, 2 anonymous edits

Image:OBASHI Business and IT Diagram.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Business_and_IT_Diagram.jpg *License*: GNU Free Documentation License *Contributors*: Stroma Software (UK) Ltd.

File:OBASHI Owner element.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Owner_element.jpg *License*: GNU Free Documentation License *Contributors*: Fergus Cloughley

Image:OBASHI Business Element.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Business_Element.jpg *License*: GNU Free Documentation License *Contributors*: Fergus Cloughley

Image:OBASHI Application Element.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Application_Element.jpg *License*: GNU Free Documentation License *Contributors*: Fergus Cloughley

Image:OBASHI System Element.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_System_Element.jpg *License*: GNU Free Documentation License *Contributors*: Fergus Cloughley

Image:OBASHI Hardware Element.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Hardware_Element.jpg *License*: GNU Free Documentation License *Contributors*: Fergus Cloughley

Image:OBASHI Infrastructure Element.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Infrastructure_Element.jpg *License*: GNU Free Documentation License *Contributors*: Fergus Cloughley

Image:Stroma Business and IT Diagram.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Stroma_Business_and_IT_Diagram.jpg *License*: Creative Commons Attribution-Sharealike 3.0 *Contributors*: © Stroma Software UK

Image:OBASHI Dataflow Analysis View.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:OBASHI_Dataflow_Analysis_View.jpg *License*: Creative Commons Attribution-Sharealike 3.0 *Contributors*: © Stroma Software (UK) Limited

File:The Zachman Framework of Enterprise Architecture.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:The_Zachman_Framework_of_Enterprise_Architecture.jpg *License*: Creative Commons Attribution-Sharealike 3.0 *Contributors*: User:Ideasintegration, User:SunSw0rd

File:Zachman Frameworks Collage.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Zachman_Frameworks_Collage.jpg *License*: GNU Free Documentation License *Contributors*: Marcel Douwe Dekker

File:Zachman Framework Detailed.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Zachman_Framework_Detailed.jpg *License*: GNU Free Documentation License *Contributors*: Marcel Douwe Dekker based on earlier work of Phogg2 et al.

File:Simplification Zachman Enterprise Framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Simplification_Zachman_Enterprise_Framework.jpg *License*: unknown *Contributors*: Al Zuech, Director, Enterprise Architecture Service at the US Department of Veterans Affairs.

File:Zachman Framework Model.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:Zachman_Framework_Model.svg *License*: Creative Commons Attribution 3.0 *Contributors*: Marcel Douwe Dekker (image) + SunSw0rd (text)

File:Example of Zachman Framework Rules.JPG *Source*: http://en.wikipedia.org/w/index.php?title=File:Example_of_Zachman_Framework_Rules.JPG *License*: unknown *Contributors*: Al Zuech, Acting Chief Architect

File:TEAF Matrix of Views and Perspectives.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:TEAF_Matrix_of_Views_and_Perspectives.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

File:Framework for EA Direction, Description, and Accomplishment Overview.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Framework_for_EA_Direction,_Description,_and_Accomplishment_Overview.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

File:TEAF Products.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:TEAF_Products.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

File:TEAF Work Products for EA Direction, Description, and Accomplishment.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:TEAF_Work_Products_for_EA_Direction,_Description,_and_Accomplishment.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

File:EAP mapped to the Zachman Framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:EAP_mapped_to_the_Zachman_Framework.jpg *License*: Public Domain *Contributors*: The Chief Information Officers Council

File:DOD C4ISR Architecture Framework Products Mapped.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DOD_C4ISR_Architecture_Framework_Products_Mapped.jpg *License*: Public Domain *Contributors*: Rob Thomas II

File:DoD Products Map to the Zachman Framework Cells.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoD_Products_Map_to_the_Zachman_Framework_Cells.jpg *License*: Public Domain *Contributors*: P.Kathleen Sowell, The MITRE Corporation (original author;slide has been publicly released & widely used throughout Government)

File:DoDAF Support to the Builder.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF_Support_to_the_Builder.jpg *License*: Public Domain *Contributors*: DoD

File:LISI Reference Model 1997.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:LISI_Reference_Model_1997.jpg *License*: Public Domain *Contributors*: Department of Defense, the Integrated Architectures Panel of the C4ISR Integration Task Force.

File:DOE Information Architecture Conceptual Model.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DOE_Information_Architecture_Conceptual_Model.jpg *License*: Public Domain *Contributors*: US DOE

File:DoDAF Perspectives and Decomposition Levels.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF_Perspectives_and_Decomposition_Levels.jpg *License*: Public Domain *Contributors*: DoD Architecture Framework Working Group

File:Integrated Process Flow for VA IT Projects.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Integrated_Process_Flow_for_VA_IT_Projects.jpg *License*: unknown *Contributors*: Dr. John A. Gauss, Assistant Secretary for Information and Technology, va.gov.

File:VA Zachman Framework Portal.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:VA_Zachman_Framework_Portal.jpg *License*: unknown *Contributors*: Al Zuech, Director, Enterprise Architecture Service at the US Department of Veterans Affairs.

File:VA EA Repository Introduction.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:VA_EA_Repository_Introduction.jpg *License*: unknown *Contributors*: va.gov

File:A Tutorial on the Zachman Architecture Framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:A_Tutorial_on_the_Zachman_Architecture_Framework.jpg *License*: unknown *Contributors*: va.gov

File:VA EA Meta-Model Cell Details Enlarged.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:VA_EA_Meta-Model_Cell_Details_Enlarged.jpg *License*: unknown *Contributors*: Albin Martin Zuech, Director, VA Enterprise Architecture Service (Retired)

File:DoD Architecture Framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoD_Architecture_Framework.jpg *License*: Public Domain *Contributors*: DoD

File:DoDAF Evolution.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF_Evolution.jpg *License*: Public Domain *Contributors*: DoD

File:DoDAF Linkages Among Views.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF_Linkages_Among_Views.jpg *License*: Public Domain *Contributors*: DoD

File:DoDAF-V2.0-Viewpoints2.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF-V2.0-Viewpoints2.jpg *License*: Public Domain *Contributors*: DoD

File:DoDAF-V2.0-Viewpoints3.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF-V2.0-Viewpoints3.jpg *License*: Public Domain *Contributors*: DoD

File:DoDAF-V2.0-Viewpoints.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoDAF-V2.0-Viewpoints.jpg *License*: Public Domain *Contributors*: DoD

File:Integrated Architecture.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Integrated_Architecture.jpg *License*: Public Domain *Contributors*: DoD

File:NR-KPP-ProductsMatrix.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:NR-KPP-ProductsMatrix.jpg *License*: Public Domain *Contributors*: DoD

Image:FDIC's Enterprise Architecture Framework.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:FDIC's_Enterprise_Architecture_Framework.jpg *License*: Public Domain *Contributors*: OIG

Image:Self-Funding Model for Future IT Development.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Self-Funding_Model_for_Future_IT_Development.jpg *License*: Public Domain *Contributors*: CIO Council

Image:Five-Year Technology Roadmap.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Five-Year_Technology_Roadmap.jpg *License*: Public Domain *Contributors*: CIO Council

Image:Structure of the FEAF Components.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Structure_of_the_FEAF_Components.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

Image:Performance Reference Model.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Performance_Reference_Model.jpg *License*: Public Domain *Contributors*: FEA

Image:BRM Overview.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:BRM_Overview.jpg *License*: Public Domain *Contributors*: FEA

Image:Service Component Reference Model.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Service_Component_Reference_Model.jpg *License*: Public Domain *Contributors*: FEA

Image:DRM Collaboration Process.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DRM_Collaboration_Process.jpg *License*: Public Domain *Contributors*: FEA

Image:Technical Reference Model.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Technical_Reference_Model.jpg *License*: Public Domain *Contributors*: FEA

Image:Architectural Levels and Attributes.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Architectural_Levels_and_Attributes.jpg *License*: Public Domain *Contributors*: Federal Enterprise Architecture Program Management Office, OMB

File:DoE Information Architecture View.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoE_Information_Architecture_View.jpg *License*: Public Domain *Contributors*: Federal Aviation Administration

Image:Treasury Enterprise Architecture Framework concept.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Treasury_Enterprise_Architecture_Framework_concept.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

Image:Blueprint Roadmap to Treasury IT Modernization.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Blueprint_Roadmap_to_Treasury_IT_Modernization.jpg *License*: Public Domain *Contributors*: Standards and Configuration Management Team (SCMT) of the Treasury Enterprise Architecture Sub-Council (TEAC)

Image:EA Development Environment.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:EA_Development_Environment.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

Image:Framework for EA Direction, Description, and Accomplishment Overview.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Framework_for_EA_Direction,_Description,_and_Accomplishment_Overview.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

Image:TEAF Matrix of Views and Perspectives.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:TEAF_Matrix_of_Views_and_Perspectives.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

Image:Enterprise Life Cycle activities.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Enterprise_Life_Cycle_activities.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

Image:TEAF Products.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:TEAF_Products.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

File:System Interface Description, Levels 1, 2, 3, 4—Generic Examples.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:System_Interface_Description,_Levels_1,_2,_3,_4—Generic_Examples.jpg *License*: Public Domain *Contributors*: Department of the Treasury Chief Information Officer Council

Image:Enterprise Life Cycle.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Enterprise_Life_Cycle.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

File:Enterprise Architecture Process.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Enterprise_Architecture_Process.jpg *License*: Public Domain *Contributors*: Chief Information Officer Council

Image:DoD Architecture Life Cycle.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:DoD_Architecture_Life_Cycle.jpg *License*: Public Domain *Contributors*: DoD Architecture Framework Working Group

Image:Enterprise Performance Life Cycle.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Enterprise_Performance_Life_Cycle.jpg *License*: Public Domain *Contributors*: U.S. Department of Health & Human Services

File:SDLC - Software Development Life Cycle.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:SDLC_-_Software_Development_Life_Cycle.jpg *License*: Creative Commons Attribution-Sharealike 3.0 *Contributors*: User:Cliffydcw

File:CPT-SystemLifeSycle.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:CPT-SystemLifeSycle.svg *License*: Creative Commons Zero *Contributors*: Pluke

Image:Systems Development Life Cycle.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:Systems_Development_Life_Cycle.jpg *License*: Public Domain *Contributors*: US Department of Justice (redrawn by Eugene Vincent Tantog)

Image:SDLC Phases Related to Management Controls.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:SDLC_Phases_Related_to_Management_Controls.jpg *License*: Public Domain *Contributors*: U.S. House of Representatives

Image:SDLC Work Breakdown Structure.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:SDLC_Work_Breakdown_Structure.jpg *License*: Public Domain *Contributors*: U.S. House of Representatives

Image:Tecnology Life Cycle.png *Source*: http://en.wikipedia.org/w/index.php?title=File:Tecnology_Life_Cycle.png *License*: GNU Free Documentation License *Contributors*: Cirt, Hermitage17, Jmarchn, Ma-Lik, Mdd, Solbris, WikipediaMaster, 1 anonymous edits

Image:Process and data modeling.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:Process_and_data_modeling.svg *License*: Public Domain *Contributors*: Process_and_data_modeling.jpg: Paul R. Smith. Redrawn by Marcel Douwe Dekker derivative work: Razorbliss (talk)

Image:IDEF Diagram Example.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:IDEF_Diagram_Example.jpg *License*: Public Domain *Contributors*: Defense Acquisition University

Image:4-3 Data Modelling Today.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:4-3_Data_Modelling_Today.svg *License*: Creative Commons Attribution-Sharealike 3.0,2.5,2.0,1.0 *Contributors*: 4-3_Data_Modelling_Today.jpg: Matthew West and Julian Fowler derivative work: Razorbliss (talk)

Image:BPMN-AProcesswithNormalFlow.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:BPMN-AProcesswithNormalFlow.svg *License*: Public Domain *Contributors*: BPMN-AProcesswithNormalFlow.jpg: Tttt1234 derivative work: Hazmat2 (talk)

Image:Government Business Reference Model.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:Government_Business_Reference_Model.svg *License*: Public Domain *Contributors*: FEA

Image:Islm.svg *Source*: http://en.wikipedia.org/w/index.php?title=File:Islm.svg *License*: GNU Free Documentation License *Contributors*: Thomas Steiner

Image:James Webb Primary Mirror.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:James_Webb_Primary_Mirror.jpg *License*: Public Domain *Contributors*: Originalwana

File:US Navy 080111-N-8273J-033 Chief of Naval Operations (CNO) Adm. Gary Roughead talks with project managers while touring Pacific Beacon, the Navy's first large-scale housing privatization facility for single Sailors.jpg *Source*: http://en.wikipedia.org/w/index.php?title=File:US_Navy_080111-N-8273J-033_Chief_of_Naval_Operations_(CNO)_Adm._Gary_Roughead_talks_with_project_managers_while_touring_Pacific_Beacon,_the_Navy's_firs *License*: Public Domain *Contributors*: -

License