NISTIR 5522

Information Technology Engineering and Measurement Model:

Adding lane markings to the information superhighway

Marvin Zelkowitz
Computer Systems Laboratory
National Institute of Standards
and Technology
&
Computer Science Department
University of Maryland
College Park, MD 20742

Barbara Cuthill
Computer Systems Laboratory

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Gaithersburg, MD 20899

November 1994

NIST

NISTIR 5522

Information Technology Engineering and Measurement Model:

Adding lane markings to the information superhighway

Marvin Zelkowitz
Computer Systems Laboratory
National Institute of Standards
and Technology
&
Computer Science Department
University of Maryland
College Park, MD 20742

Barbara Cuthill
Computer Systems Laboratory

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards
and Technology
Gaithersburg, MD 20899

November 1994

U.S. DEPARTMENT OF COMMERCE
Ronald H. Brown, Secretary

TECHNOLOGY ADMINISTRATION
Mary L. Good, Under Secretary for Technology

NATIONAL INSTITUTE OF STANDARDS
AND TECHNOLOGY
Arati Prabhakar, Director

Executive Summary

Technological changes in the telecommuncations, broadcasting and information technology industries are bringing about vast changes in our economic landscape. How industry and government adapt to these changes will greatly affect the economic climate as the United States enters the next century. Understanding the roles that modern computer and communication technology plays in developing the electronic marketplace upon which the U.S. economy is increasingly dependent is important in planning future technical and economic developments.

The growth of high capacity international networks linking together millions of computers is the major impetus for these changes. Whereas commerce previously followed river or highway routes with marketplaces arising out of interconnections among several of these routes, in the future, commerce will be along these electronic information routes, the so-called "information superhighway." How to build such information networks, what the policies are regulating their use and how policies will foster a "level playing field" for equitable access to electronic commerce are all issues that need to be addressed.

Issues can be classified under three general concerns:

1. *Economic issues.* The development of the electronic marketplace must be considered. Electronic purchasing, payments, advertising, delivery, etc. are all issues that must be resolved. How the consumer interacts with the supplier must be considered.

2. *Policy issues.* Policies on access, development and use of the electronic marketplace are needed. Issues such as public access, fair access, monopolies, public utilities versus independent suppliers, and competition from alternative sources have to be developed.

3. *Technological issues.* How to build this information superhighway depends upon present and future technology. As this report shows, this technology will grow out of the current information technology, telecommunications and broadcasting industries.

This report is not meant to address the economic, policy or political issues concerning these developments. The role of government and industry in developing the information superhighway is certainly an important concern; however, the major goal of this report is to describe a structure for the underlying technology that may be useful in developing the appropriate economic and policy models.

Having an appropriate technological base is a prerequisite for addressing the economic and policy issues needed to use this technology. The confluence of the telecommunications, broadcasting, and information technology industries is driving these changes. In particular this report discusses a model, called the Information Technology Engineering and Measurement Model (ITEM Model), which can be used to describe the information technology base used by organizations as they grapple with the issues in developing the electronic marketplace of the next century.

The ITEM model represents the processes and automation present at several levels in an organization. Using it, technology used to develop the various models that drive the business decisions of the organization can be described. In this report the ITEM model is presented with two examples of its use: (1) To describe the software engineering activities within an organization and (2) To describe the information flow within an entire enterprise.

Table of Contents

1. Describing the National Information Infrastructure: The ITEM Model

Abstract

Development of the National Information Infrastructure (NII) will depend upon building vast networks of interconnected computers communicating over high-speed digital lines. This paper describes the growth of the NII concept and proposes a model - the Information Technology Engineering and Measurement (ITEM) model - that may be useful in describing the set of services an operational NII may contain.

1.1 Introduction

The confluence of the data-intensive telecommunications, broadcasting and information technology industries has led to various proposals to create a National Information Infrastructure (NII), the so called "information superhighway," to allow each of these compatible technologies to evolve [15]. In order to best guide the development of the NII, a model of information usage in an enterprise will permit industry to improve their information processing infrastructure using a common nomenclature and set of services upon which to describe their developments. In this paper such a model, built upon previous work in software engineering environments, may serve as a mechanism to guide the technological developments of the NII.

The following subsection first describes the need for the NII, its possible overall structure, and then presents the description of this model.

1.1.1 Information Infrastructure

As used in the NII, "infrastructure" is analogous to the set of services that are ubiquitious in our society today. For those living in most urban areas, highways, railroads, and airlines provide transportation to almost any other place in the United States. When purchasing homes, buyers expect to have electricity, water, natural gas and sewer services available without having to make special preparations for their installation.

Within the last 50 years, three new services have or are starting to achieve this ubiquitous state – telecommunications, broadcasting, and information technology:

Telecommunications. For perhaps the last 60 years, telephone service has been widely available, but our notion of this industry has changed with the technology. Today, while telephones still use 2,500 Hz analog signals for communication with other telephones, most of the internal telephone network has been digitized with gigabit-rate fiber optics and satellite transmission media. The telephone industry is rapidly providing new services using these new digital technologies. Voice mail, call forwarding, call waiting, and fax transmission are some examples of services now available supplementing the simple call-and-talk model of the telephone. Numerous products are now being sold using the telephone as a computer terminal with the touch keypad as an input device and the earpiece as an output device.

Broadcasting. The broadcasting industry began in 1922 with radio, developed after World War II with television and changed during the 1960s and 1970s with cable extending over-the-air

transmissions with a piped-in service. Today, most urban areas have cable service with 30 to 50 channels of 6 mHz signals, providing the equivalent of several gigabits per second transmission, entering most homes. The cable industry is exploring new uses for this bandwidth including video on demand, home purchasing via television as an extension to the current "home shopping" television channels, interactive video from a central library of such services, and full information retrieval from sources such as the public library.

Information technology. Concepts like the Internet linking together millions of world-wide computers have captured the fancy of the public – or at least journalists who like to write about it. Services like *Gopher* and the *World Wide Web* provide quick access to data repositories world-wide. Private companies[1], like Prodigy, Compuserve, MCI, America-On-Line and others, all provide similar capabilities allowing individuals to reach large computer-based information resources.

The current information technology industry is built upon the present telecommunications industry as a value-added product. The Internet is linked together with high-speed digital lines leased from telecommunications companies. The home computer is linked into these networks via modems using lines from the local telephone companies; however, this local loop is inefficient. Current technology (i.e., operating at 20k bits per second) converts data from a home computer into a 2.5 kHz analog signal, sends it to the local telephone office which converts it into a 64k bit per second digital signal and transmits it to the local office of the receiving computer, which converts it into an analog signal that is sent to the receiving home where is is reconverted into a 20k bit per second digital stream for the receiving computer. A 64k bit computer-to-computer link would be much more efficient. What is worse, information theory tells us that with a 2.5 kHz voice signal, the maximum bit rate achievable is on the order of 30,000 bits per second.[2] Thus increases in modem speed from the 110 bps "teletype standard" of 1970 to 1200, to 2400, to 9600, to today's 38,400 bps will not continue for long. The information technology industry is looking at ways to increase this transmission speed and to provide additional services beyond those currently available. For example, experiments with MBONE (Multicast Backbone) television transmission over the Internet show the feasibility of low-bandwidth video, but still require higher bandwidth than available on most home telephone circuits.

Already, information technology has greatly transformed the marketplace. Protocols such as Electronic Funds Transfer (EFT) have greatly affected the flow of dollars from consumer to producer and from employer to employee. Electronic Data Interchange (EDI) greatly affects the ordering and flow of components from supplier to manufacturer. The speed and accuracy of such transfers makes techniques such as Just-In-Time (JIT) methods, with reduced production times needing less inventory storage, more practical and cost effective.

All three of these industries are evolving towards a common set of goals: (1) Providing "gigabit rate" digital transmission; (2) Providing ubiquitous "terminals" (e.g., telephones, televisions, home

[1] Certain commercial products are identified in this paper to specify adequately the applicability of the model. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products are necessarily the best available for the purpose

[2] Note that current modems that transmit at speeds of 38k are really transmitting only 9.6k bits per second, but achieve a higher effective speed by use of data compression which uses the redundancy within ASCII text to send 8-bit characters at an average of 1 to 3 bits each, depending upon content.

Transmission	Reliable and secure transmission of video, audio, and textual data throughout the enterprise.
Mid-level Services	Creation of services to support voice communication, video communication, textual communication, data repositories, process management, information and infrastructure security and other related services.
Applications	Development of end-user enterprises by the private sector. Initial interest is in the following seven industries: health care, environmental monitoring, manufacturing, communications, government services, education and libraries.

Table 1.1: NII services

computers); (3) Providing interactive services; and (4) Using computer technology to drive local capabilities.

All three industries currently revolve around vertical markets. Each industry is developing capabilities to provide the full range of services from the underlying communication mechanism to the interactive value-added service. As these capabilities evolve, there will probably be a merging of these organizations with their technologies as well as a layering of service providers. Some providers will provide the communication infrastructure while others will provide the interactive capabilities.

In order for these industries to prosper, there needs to be consensus on what technologies provide these new "ubiquitious" services, what standards guide their development and interactions, and how individuals will interact with these new service providers – hence the current interest in the NII.

1.1.2 Model of the NII

One proposal for the NII is given in Table 1.1 [15]. A base-level transmission network provides the basic data-carrying capabilities over high capacity *bitways* with the ability to carry voice, audio, text, graphics, video, animation, and other data at high speeds over the same set of communications lines. Packet switching technology, such as in the Internet, must be provided for achieving reliable and efficient transmission – including real-time data. Communication protocols, such as Asynchronous Transfer Mode (ATM), have been designed for creating meassages that permit long data files, electronic mail, and real-time video and audio transmissions to coexist on the same communication path.

Mid-level services build on the data transmission services to provide information. Agents and brokers provide sets of services for the user community and transmit the information along an appropriate infrastructure bitway. For example, animation consists of approximately 30 video images per second. Protocols such as the MPEG (Motion Picture Experts Group) standards provide for compression of picture information so that only a small fraction of the 60M bits needed to describe thirty 600×400 bit full-motion video images in 256 colors need actually be transmitted per second. This vast middle ground of service (e.g., data repositories, communication among providers and users) is not well defined for the NII. It is at this level where new technology and applications are most needed.

Research area	Addresses
C.1 Network components and protocols	
C.2 Information appliances and servers	
C.3 Information access (4) Interoperation architecture	*
C.4 Multimedia information techniques	
C.5 Infrastructure for application development (1) Requirements for interoperability	*
C.6 Dependability and manageability	*
C.7 Ease of use	
C.8 Interoperability	*
C.9 Security and privacy	
C.10 Portability, mobility and ubiquity	

Table 1.2: NII Research Agenda

At the applications level are the consumer services. Initially the participants will be the current telephone, cable, and broadcasting companies refining their current set of products and services. How they provide such services as they start to use information networks will evolve. In addition, as the NII develops, additional companies, such as libraries, manufacturers, and others listed in Table 1.1 [16], will enter this arena and provide value-added services based upon the set of mid-level services that NII providers create.

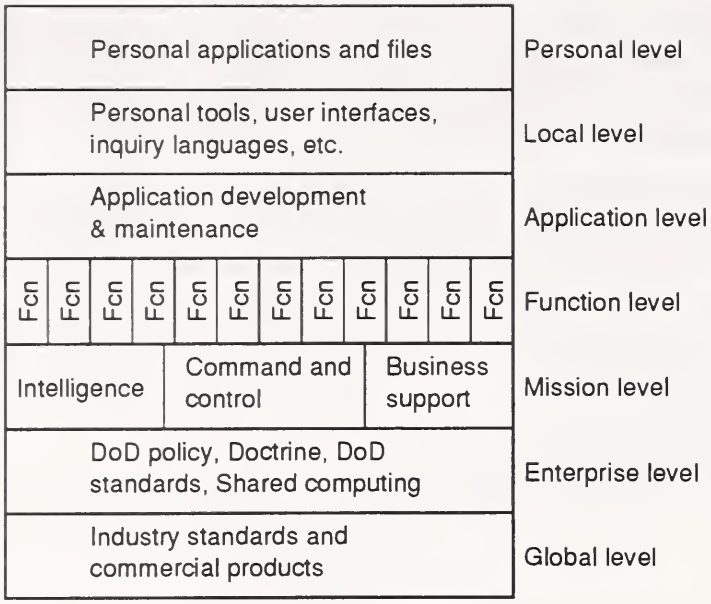
The technological question that this paper addresses is what are the mechanisms that are needed to describe the set of services provided by the various levels of the NII? Continuing like Humpty Dumpty in "Alice in Wonderland" with words meaning whatever we want them to mean is not effective.

1.1.3 NII Research Directions

There has already been considerable discussion of NII goals, technical issues and research needs [19]. The model described here represents initial efforts towards some of these. In Table 1.2 are several recommended NII research needs with an indication of which ones this model addresses. The model is general and also addresses some of the other goals in that report, although only indirectly.

1.2 Overview of the Model

The *Information Technology Engineering and Measurement* (ITEM) model is an enterprise-wide service-based model that builds upon existing models of information technology use in organizations in order to provide a total picture of information processing needs. *Enterprise-wide* means the major activities that drive the information management decisions of a large organization, such as a corporation or government agency. "An enterprise model is the essential ingredient of any architectural approach. This model shows both the data needed by the entire organization and the processes which manipulate that data" [25]. The ITEM builds upon the following models:



(Read up, ITEM model reads down)

Figure 1.1: CIM Integration Architecture Model

1. The National Institute of Standards and Technology (NIST) and European Computer Manufacturers Association (ECMA) Software Engineering Environment (SEE) framework reference model [17] provides a base-level infrastructure set of services needed to support environment functionality.

2. The Project Support Environment Standards Working Group (PSESWG) Project Support Environment (PSE) reference model [6] extends the framework services to a set of end-user services providing tasks needed in the software engineering development domain.

3. The POSIX.0 open system environment reference model [10] provides a model of communication and interaction among environment components.

4. The Corporate Information Management (CIM) interface architecture model [7] (See Fig. 1.1) provides a hierarchical view of services within an enterprise.

5. The Quality Improvement Paradigm (QIP) model of Basili [3] emphasizes the process, the product and measurement as primary drivers of an organization.

While the CIM structure represents a static view of the information activities of an enterprise, the ITEM model appends four stimuli that influence an organization's information use (Fig. 1.2):

- *Market forces* are external stimuli which affect the organization. These include consumer demand, resource availability, or government regulation.

- *Technological changes* are the information resources available to the organization to solve the problems defined by the above market forces. It is the ability of new technology to disrupt the standard business mechanisms with new methods that drives the organization forward [9].

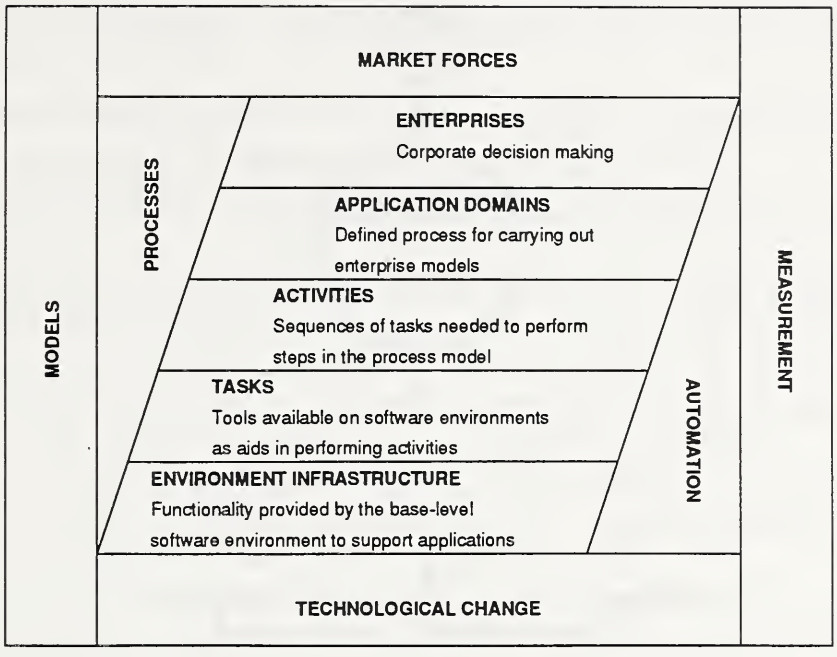


Figure 1.2: ITEM Model

- *Models* are the abstractions needed to understand how the available technology can address changes imposed by market forces.
- *Measurement* determines how well the enterprise's information processes match the abstractions present in the models.

The relationship among these four stimuli are given in Fig. 1.3. An organization uses the models to develop measures of the effects of market forces and technological changes on the organization. Using these four components, the enterprise develops, tailors, and adopts strategies. The ITEM model views strategies as combinations of processes and the automated support for those processes.

The model can be viewed as a specialization of the Quality Improvement Paradigm (QIP) model of Basili [3] which emphasizes the process, the product and measurement as primary drivers of an organization. The different modeling levels (to be described next) describe the *processes* that the organization will use (e.g., the way the enterprise does its business) and the *automation* that the organization uses (e.g., the way in which it uses software and hardware information system components).

1.2.1 ITEM Model Levels

An enterprise needs to understand its information processing needs at five levels of specificity:

1. *Enterprise.* This level defines organization policy and decision making. Major corporate decisions on organization policies are made at this level with little direct concern about information technology. These decisions include the selection of corporate strategies for manufacturing capabilities, outsourcing, and product development. Measures of success would include business metrics such as profit, market share, return on investment, etc.

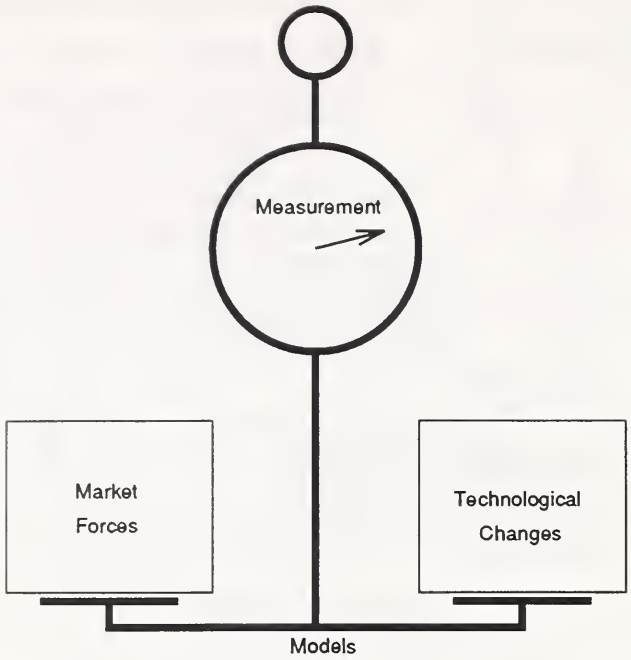


Figure 1.3: Role of model components

2. *Application domain.* Application domains are concerned about the methods for implementing enterprise decisions. Examples of application domains include software development, personnel management, and manufacturing a product line.

3. *Activities.* Activities are sequences of steps needed to address enactment of services within an application domain. Activities represent complex series of interactions that aid in solving the problems within the application domain, as will be explained below.

4. *Tasks.* Tasks represent single steps needed to carry out an activity. For example, an interactive video application from a cable TV company might be an activity consisting of the following tasks:

   - Consumer orders video.
   - Cable company schedules video on cable channel.
   - Video is transmitted.
   - Customer is billed for service.

   Each of these individual actions may be carried out using different methods by different enterprise components.

5. *Infrastructure.* Infrastructure represents indivisible components within an enterprise. Hardware technology like typewriters, fax machines, and telephones are infrastructure components. With computer software, basic functionality (e.g., data base access, communication processes, etc.) represent infrastructure.

   For example, the task of "Consumer orders video" may be enacted by invoking the following infrastructure services:

- Access to data repository to check current credit status and other billing information of consumer.
- Use of user interface services to interrogate consumer for video desired.
- Access to respository to check availability of requested video.
- Invoke process services to pass consumer and video information to scheduling task.

The divisions separating these levels are somewhat arbitrary. A task is a single step, an activity is a set of such tasks and an application domain is a set of activities solving a larger problem. For example, although software quality assurance can be an activity (as is done in [6]). Could it be a task or be an application domain in a different structure for this classification scheme? Can an application domain (e.g., software engineering) be considered an activity in a larger application domain (e.g., product development)? The answer is probably yes. In what follows tasks will be sets of actions supported by a single tool or small tool set and application domains as larger product-oriented sets of activities that achieve organizational goals.

Over time a process may change in its relative complexity and level. Increased automation may simplify an activity so much that it becomes the task of running one tool. For example, generating and testing the code for a graphical user interface (GUI) now can consist of running one GUI building tool rather than using an editor, compiler, linker, debugger, and simulator to write, debug and test the code. Alternatively the enterprise may place an increased emphasis on some area requiring the elaboration of previously simple tasks into activities. For example, an enterprise-wide increased emphasis on quality may lead to the use of more testing tools and more elaborate recording and analysis of the products of those tools. Infrastructure, tasks, and activities are changing over time in response to changes in the market place and available technology, while application domains and enterprises are more stable. What constitutes a complex activity or application domain may become simpler with increased automation or a simple task may become more complex with increased elaboration.

It should be made clear that the levels of the ITEM model are logical and not physical boundaries. Notations, such as IDEF0 [14], are able to model the relationship among infrastructure, tasks, activities and application domains. Although business plan decisions at the enterprise level may not involve automation, the process model to define, implement, and monitor the plan may require considerable automation – data bases to store information, word processors to produce reports, spreadsheets for producing "what if" scenarios, decision support expert systems for analyzing strategies, etc.

1.2.2 Automation versus Process

Each level is concerned about the degree of automation and the set of processes needed to carry out the specifications of that level. At the infrastructure level, automation, also termed *enabling technology* [9], is prevalent with little concern for specific behaviors in an application domain. For example, a data base system, a word processor, or a spreadsheet are software technologies that are generally independent of the specifics of any application domain. Templates or schemas tailor a database to the application domain by defining the structure of its data. Thus a database provides a high degree of independent automation with little attempt to influence the behavior of the user. This is comparable to the role of the telephone, fax machine, or typewriter.

As one goes up the levels of the ITEM model, this balance changes. For example, at the highest level (e.g., determining the business plan of the enterprise), the role of automation is relatively small with the major goal to define the process that will be the most likely to increase profits, increase market share, or develop quality products. Automation comes into play as one implements the defined models as more detailed sets of activities and tasks needing or utilizing automation support.

An alternative way to view this hierarchy is in the translation of data to information to knowledge. Infrastructure services (e.g., a word processing program, a telephone, a fax machine) process data without much interpretation in its use. At intermediate levels, the basic data is converted to information (e.g., collecting data on software development to determine product reliability, productivity, profitability). With this information, knowledge can be extracted (e.g., Is the waterfall life cycle more effective than the spiral model? Does concurrent engineering improve development attributes?) At present, automation effectively collects data, and individuals are needed to analyze that data and to process knowledge. Much current research in computer technology (e.g., AI research) is in developing methods for extracting information from data and in analyzing information to determine knowledge.

Enactment of an activity requires a combination of both process and automation support and may successfully be solved by various combinations of both. For example, the activity of sending a message to an individual at another location within the enterprise can be solved in multiple ways:

1. *Write out information on a sheet of paper and deliver it to recepient.* No technology is needed, other than a transportation mechanism to get to other location. This can be viewed as almost a pure process solution to the problem.

2. *Write out information on a sheet of paper and send fax.* This method uses no computer support and only minimal automation support (e.g., use of a fax machine).

3. *Use a word processor to develop memo, print it, and then send copy via fax machine.* This uses a minimal amount of automation in developing the memo, and there is no knowledge integration step which combines the memo generation process (e.g., use of word processor) with the transmission process (e.g., use of fax machine).

4. *Use a word processor to develop memo, and then use a command such as "send fax" to automatically have computer-installed fax hardware send the memo to the receiving site.* In this case, the word processor is integrated with the fax machine and automation support for this activity is very high.

5. *Use word processor and "send fax" software to send fax directly to receiving computer.* In this case, no fax machine is used at all, and a direct computer-to-computer link is established.

6. *Use a word processor and then send electronic mail to recepient.* This avoids the concept of a fax machine totally, and represents a total computer solution to the problem.

In all six cases the results are the same – the receiver at another location gets a memo; however, the process and the method of automation for achieving the goal differ. For example, solutions two through five (i.k., use of fax transmission) depend upon the telephone network to provide telecommunications support and the use of fax protocols for secure and correct transmission of the message. The electronic mail solution, while eliminating the direct use of telephone lines and fax

protocols, assumes the existence of a more complex communications network linking the computers together (which may use telephone lines).

1.3 Sample use of Model

Consider the following extension to the typical cable-TV service. In this example, assume that the consumer wishes to view a movie. The model presented here is that the cable service provides a "virtual VCR" to the user.

Enterprise. The enterprise here is the cable broadcasting company which provides video services to its customers.

Application domain. The application domain is video projection – the process of sending a stream of video images, about 30 per second, to the consumer's monitor.

Activities. The set of activities models the typical functions of using a home VCR: Rent video; Insert video (i.e., load VCR), Play video, Pause (i.e., suspend for limited time), Halt (i.e., unload VCR), Billing (i.e., pay for movie).

Tasks. The tasks provided by the NII include: Send video images at 30 per second, Suspend transmission, Data repository updates to consumer data base. Note that activities are described in application domain terms – concepts related to the "virtual VCR" being modelled. Tasks, however, start to sound application independent – transmit, data base update, etc.

Infrastructure. This provides reliable transmission of video images as well as reliable data base accessing for ordering and billing purposes. Real-time response of about 30 milliseconds per image is crucial for achieving smooth motion video.

Today, there are many existing enterprises offering a variety of services packaged in various applications. How they support those services will change as the infrastructure evolves. As this new infrastructure develops, new activities, such as the above interactive video example, will appear as service suppliers learn how to effectively use and exploit this new technology.

The tasks and activities represent mid-level services of the NII. This is clearly an area that needs research and development. For example, it is not clear if these levels represent the cable broadcast company as a mid-level service supplier or if the cable company is an application domain service that uses mid-level communication services provided by others. Solution of this question is both a technological issue of how to build this structure as well as an economic and policy decision of what sort of infrastructure is to be created and how is it both controlled and managed.

Given this structure, it would be easy for a library, for example, to define an application domain to provide on-line reference material using the same or similar set of activities and tasks necessary to provide for video transmission.

1.4 Environment Evolution

Organizations, like individuals, undergo change. Therefore, an effective information technology model of an enterprise must reflect the changes that organizations undergo as they modify their information technology usage over time.

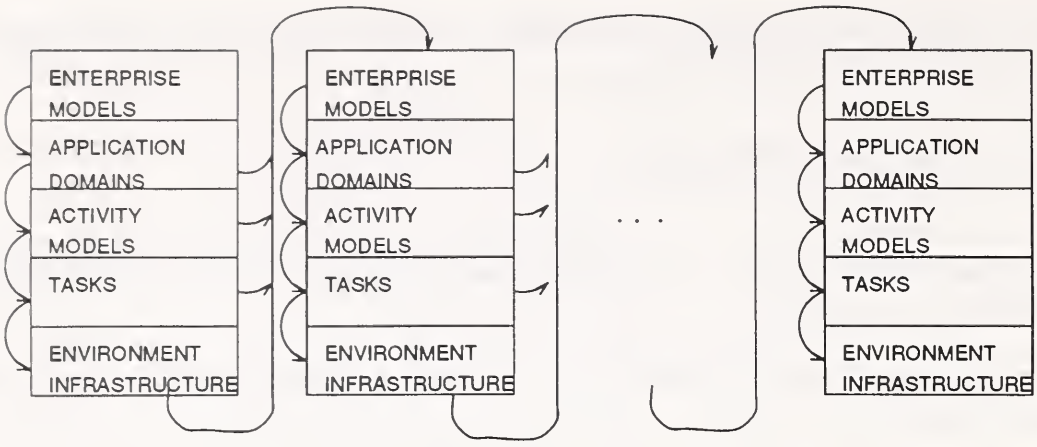


Figure 1.4: Evolution of the enterprise model over time

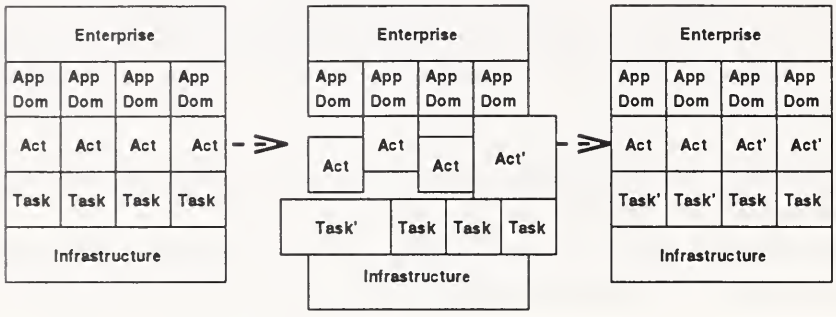


Figure 1.5: Current enterprise evolution

1.4.1 Enterprise Evolution

An enterprise's computer resource options change as computer performance, price, and functionality change. This constantly affects the policy decisions an organization must make to remain competitive – or in the case of government – relevant. Fig. 1.4 describes this process.

1. The Enterprise determines what application domains need support.
2. The application domain determines which activities need support.
3. The activity determines which tasks to implement.
4. The set of tasks determine what environment infrastructure is needed.
5. The environment infrastructure determines what sort of information can be processed. This in turn affects what the Enterprise can be expected to obtain. This process feeds information back to the enterprise level which drives further changes in the process models.

As shown in Fig. 1.2, external stimuli affect this process. One level may become out of balance with others and affect the entire hierarchy. For example, the rapid decline in workstation prices in the 1980s greatly affected how organizations viewed their computer resources and the supply of "cheap cpu cycles" led to a rapid increase in corporate use of the desktop PC.

This ideally becomes an iterative process that drives the enterprise forward. Unfortunately, reality is rarely so formal or regular. Fig. 1.5 represents a more typical scenario. An organization has an established set of application domains, activities and tasks. Initially, the enterprise modifies one activity (i.e., changing how the organization tests software) and one task (e.g., aquire a new database). This is represented as the center figure of Fig. 1.5 which shows that these changes affect many other activities and tasks. Given the new capabilities of the modified activity and task, several other activities and tasks are modified to address these changes and the enterprise settles down into a new equilibrium set of processes – at least until the next set of changes. The next chapter describes one organization's efforts at evolving its environment due to required changes in its information systems.

1.4.2 Enterprise Devolution

Ideally, information flows from one level of the model to drive decisions at the next lower level, and lower level services support the processes of the higher level. Information flowing down will tend to refocus the level below. For example, if the enterprise adds a goal of lowering maintenance costs by improving software quality, new processes will have to be developed. But what if that goal is ignored? Alternatively information flowing up from the infrastructure and task levels can be corrupted causing a misunderstanding of the technical capabilities available to the enterprise.

Enterprise devolution can occur if the levels of the model become inconsistent or evolve in different directions. For this to happen the feedback process encouraging consistent evolution will have failed. There are several explanations for that failure:

- Retention of unneeded processes supported by activities or tasks when the goals they address have been eliminated
- Retention of unneeded constraints after the technology imposing the constraint has improved
- Changes in the enterprise process models or goals that fail to propagate
- Changes in the external environment affecting the implementation of the enterprise model
- Imposition of enterprise goals that are unobtainable with current technology
- Propagation of corrupted data either through misunderstanding or falsification

The solution to this problem is process reengineering. Process reengineering initially validates the enterprise's application domain, activities and tasks against the enterprise model and the processes chosen to support its goals. Once these relationships are understood, process reengineering can streamline the processes to match the enterprise's goals, eliminating extraneous activities and tasks while adding new processes, activities and tasks to better implement the enterprise's goals.

2. Corporate Restructuring Using the ITEM Model

Abstract

The ITEM model is used to describe the proposed changes to the information systems of one large corporation. The need to convert from a mainframe environment to a client–server model using a network of workstations is described, and the proposed set of tasks and infrastructure services needed to implement these changes is identified by the model.

The effectiveness of the ITEM model depends upon its ability to model current enterprises and the changes that enterprises undergo in their business processes. In this chapter the model is used to describe one large organization whose information system evolution was described in the literature [2].[1]

The enterprise produces cellular telecommunications hardware which is an extremely price sensitive commodity item. With prices declining, the enterprise set a target of reducing information system (e.g., computer) costs from 3% to 1% of gross revenue. This represents the major market forces on the enterprise, with the percent of gross revenue the major enterprise–level metric used to measure progress towards this goal. These costs support acquiring and developing all the software and environment infrastructure that the enterprise uses outside the systems actually embedded in its sold products. (That activity is described elsewhere [11]. The strategy for meeting that target was to convert from a mainframe and customized software environment to a less expensive client–server and commercial off–the–shelf (COTS) environment.

Just as declining prices for its products represented a market force on the enterprise, improving capabilities of *purchasing* computer products represented a technological change useful in instituting the needed changes to the enterprise.

This change in automation required substantial changes to the processes the enterprise used for supporting its computer use and minor changes to other processes throughout the organization with the installation of new tools. As shown in this chapter, changing the environment infrastructure left the environment infrastructure initially "out of balance" with respect to the rest of the enterprise and initiated a sequence of changes throughout the enterprise.

2.1 Stage 0: Initial State

This enterprise is both a manufacturer and a service provider. It manufactures a variety of computer and telecommunications equipment for an increasingly price competitive market. It also provides network services frequently as part of joint ventures. Initially, the enterprise used an environment infrastructure consistent across the entire organization with mainframes running custom-developed software and personal computers supplying some auxilliary services.

The enterprise consists of five organizational units. Each of the five produces a different product line. While there is some overlap and integration of products and markets, the five are essentially

[1]This analysis was done from information contained in the cited reference without aid from the organization itself. The purpose is to describe the use of the ITEM model, not to comment on the particular enterprise and its reengineering efforts.

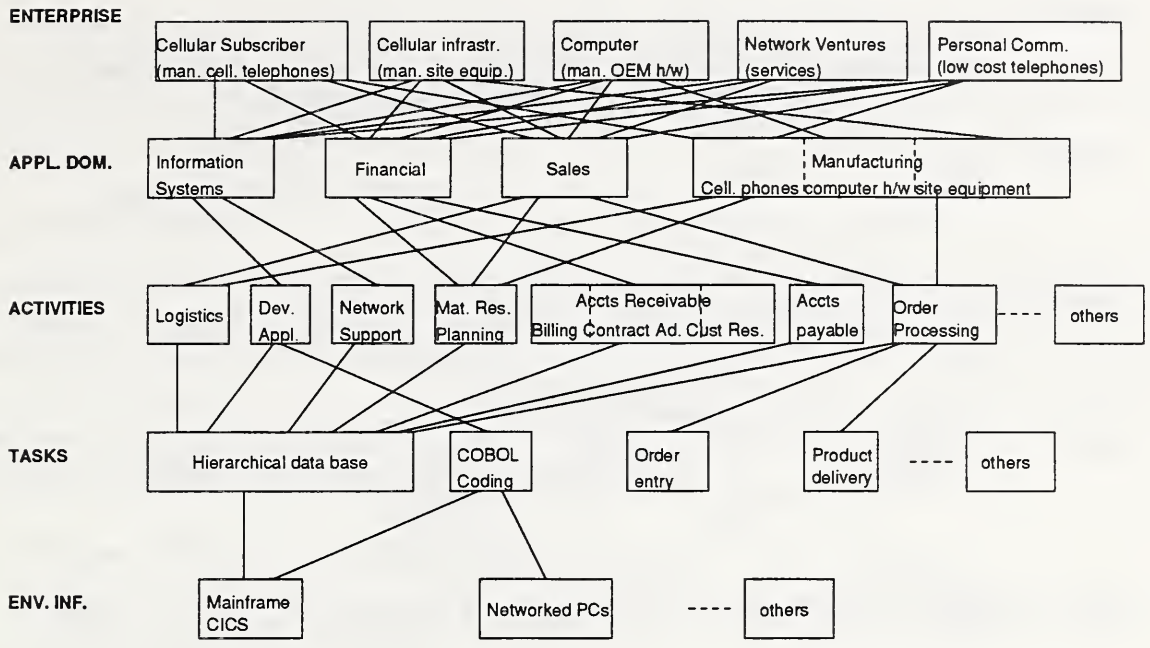


Figure 2.1: Initial state of enterprise

independent business units sharing some common business services. Each unit consists of a combination of specialized, mission critical application domains producing and marketing the specific goods it produces (e.g. cellular telephone manufacturing services) and business application domains necessary to any organization (e.g., financial services, information systems). The exact mix and relative importance of the various application domains vary for each organizational unit.

Figure 2.1 uses the ITEM model to describe this enterprise. This figure emphasizes the components of their information technology that is related to their sales activities since these were upgraded first. There are, of course, many other application domains, activities, tasks and infrastructure services needed to describe the complete organizational structure of this enterprise. Because this figure was derived from information contained in a published paper [2] and not from a detailed study of the organization, it may not be totally accurate; however, it does demonstrate the model's ability to describe such organizations. The importance of the model is that if Figure 2.1 does *not* really describe this enterprise, then such errors would be immediately apparent and changes could be made.

The initial set of changes focuses on four application domains of this enterprise: information systems, financial, sales and manufacturing. At the start of their downsizing effort, the enterprise provided predominantly centralized management information systems (MIS) application domain services to each organizational unit. MIS application development and support was a central activity with the central facility maintaining tight control over the availability of resources to each organizational unit. The enterprise also appeared to centrally administer the accounting application domain maintaining tight control over local accounting practice. On the other hand, the enterprise appeared to permit each organizational unit's manufacturing, and sales application domains to be relatively independent and more dependent on their individual product line. This may reflect the differences between mission critical and non-mission critical functions of each unit.

In the ITEM model, activities and tasks support (e.g., provide components) of application

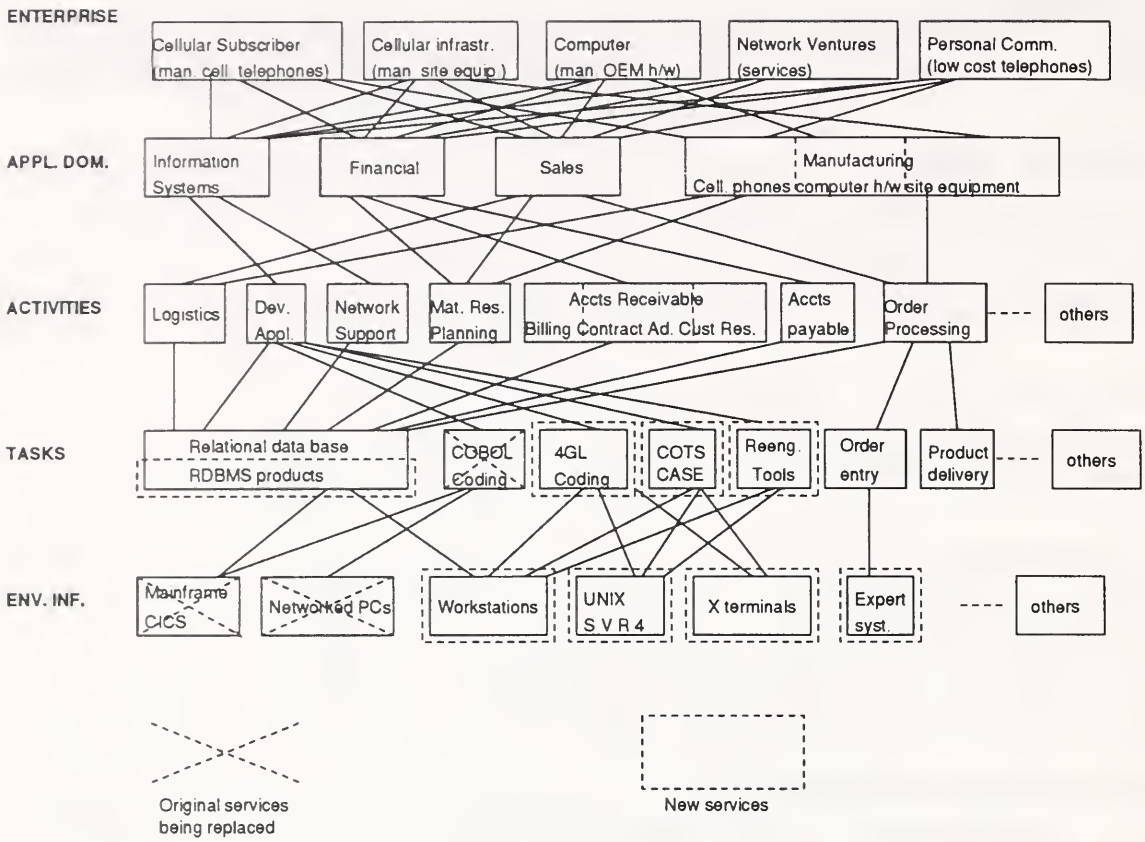


Figure 2.2: Changes to tasks and infrastructure

domain processes. Some tasks may be components of a variety of activity processes while others may be highly specialized. For example, the task of entering information into a database may be part of several activities in different application domains while the task of writing COBOL code is only part of the information systems application domain. Activities and tasks can also involve intersections of processes in different application domains. For example, an engineer may review a custom order for telecommunications hardware. This review task may be part of both a custom hardware development process and a sales process.

Figure 2.2 describes the general changes to the environment that the enterprise developed in response to its analysis of the market forces and technological changes to the enterprise. By moving to a client–server architecture the base-level infrastructure was moving away from mainframes and PCs towards a workstation and X-terminal architecture. Because of the low cost of COTS tools, there was also a movement away from custom-building of COBOL programs towards purchasing of COTS tools or building tools using CASE (Computer Assisted Software Engineering) tools and fourth generation languages (4GLs).

These changes represent alterations to the infrastructure and tasks of the enterprise. Since the business plan of the enterprise has not changed materially, the set of application domains and activities remains the same, but as Figure 2.2 demonstrates, the implementation of some of these activities changes. However, as the organization understands the capabilities of its new infrastructure, undoubtedly it will develop new activities to use these capabilities.

Any change to the tools supporting a task will have an effect on the task process which in

turn will effect the activities incorporating that task. This ripple effect migrates outward from the changed task until the rest of the application domain processes adjust to the change. The application domain itself is a sufficiently abstract view of the enterprise components that it should not be affected by a change in infrastructure.

Initial External Stimuli

There are several features described by the external stimuli affecting the organization in its initial state. The most important stimuli are the market forces bringing down the prices of the goods this enterprise produces (i.e., its income) and the technological changes providing more powerful hardware and software at a cheaper price (i.e., its costs). These stimuli encourage the organization to adopt a strategy of replacing more expensive with less expensive computer equipment while changing the processes using those computer services:

Market Forces: Exernal market forces include:

- Products that are produced are a commodity; selling price greatly affects sales; prices for its products are declining.
- Prices for computing power, in the form of UNIX servers, are declining substantially relative to mainframes.
- Custom software is expensive to maintain and offers a limited range of options.

Technological Change: Technological drivers for change include:

- Workstation capabilities are increasing; costs are decreasing.
- Good COTS software packages are available in some areas (especially in support of financial services such as accounting).
- CASE tools, 4GLs and software development processes using them are now commonly available.
- Workstation platform vendors are providing open systems.
- Methodologies using CASE tools and 4GLs are available and require smaller staff to implement.

Modeling Techniques: Modeling techniques used include:

- Enterprise uses centralized Information System Group.
- Information Systems use a traditional software development model with very little automation.
- Information Systems spends considerable resources planning next increments of computer power needed.

Metrics: Metrics used include:

- Information systems cost 3% of company revenue.

- Mainframe maintenance costs are very expensive.
- Maintenance of hardware and software requires a large staff.
- Computer power increment in client server architecture costs $500-10,000; new increment of computer power in a mainframe costs $1,000,000.

New enterprise goal

In response to the external stimuli, the enterprise set a new goal: Reduce software costs to 1% of company revenue over five years. The enterprise established several new policies and reaffirmed existing policies in line with this goal:

1. Maintain system security
2. Move to Open Systems
3. Reduce Information Systems staff by 50%
4. Maintain current level of support
5. Do not disrupt mission critical operations

Achieving these new goals, especially the last one, required a strategy of evolving to a new environment. This strategy consisted of the changes in automation and processes needed to meet the goal.

Automation Component

Changes in their automation facilities consisted of the following:

- Move from Mainframes and PCs to UNIX servers and X terminals. (Use of UNIX servers and X terminals addressed goals 2 and 3. Replacement of PCs by X terminals addressed goal 1.)
- Adopt UNIX platforms. (This addressed goal 2.)
- Move from Custom Software to COTS. (This addressed goal 3.)
- Move from non-tool supported development to CASE tools and 4GL. (This addressed goal 3.)
- Migrate all applications from Mainframe to Workstation without allowing any exceptions. (This addressed goals 3, 4, and 5.)

Process Component

Process changes required the following:

- Migrate one application at a time beginning with non-mission critical areas. (This addressed goals 4 and 5.)
- Reengineer tasks and activities to new tools.
- Change development process to heavily automated process (no coding). (This addressed goal 3.)

2.2 Stage 1: Information System Application Domain

The strategy called for implementing the change in one application domain at a time and in one organizational unit at a time. This allowed the enterprise to determine the ripple effects of changing one domain before having to change others. It is also a practical way to make substantial change in a large organization.

Changing the environment infrastructure requires changing all the tools supported by that infrastructure. This change can be replacement of the tool with a comparable COTS tool, development of an in-house equivalent or regineering the existing tool for the new platform. The strategy called for using the first choice whenever possible and considering the others in order if there were no equivalent COTS tools.

Even if a tool provides equivalent or better functionality, there will still be some change in the tasks that the tool supports. This propagation will continue changing the tasks, the activities using those tasks, and the application domain processes using those activities until the change works through the system.

The first application domain acquiring the new environment infrastructure was the information systems application domain. The new infrastructure had a substantial impact on this domain. New activities and tasks appropriate for workstations had to be developed to use the new set of 4GLs and COTS tools that were acquired and old tasks of supporting the mainframe, and using more traditional software development techniques such as COBOL programming, were discontinued.

The new set of tasks also affected other standard tasks. One example of a discarded task is that of detailed analysis of system requirements for upgrades. With upgrade increments of workstations costing between $5000 and $10000 rather than $1,000,000 for a mainframe upgrade, the information systems group no longer had to do the same planning and justification to get the next increment of computer power. (No sense in spending $100,000 for a planning study for a $10,000 upgrade.)

The new infrastructure and corporate policies also radically changed the software development process which is part of this application domain. The enterprise defined three approaches for migrating tools: purchase COTS, develop in 4GL, and reengineer old applications. Each of these new activities required substantial definition of new tasks and the purchase of new tools to support those tasks. To purchase COTS tools for other parts of the organization required tasks in business analysis and process reengineering to determine what type of tools the enterprise needed and tasks in tool evaluation to decide what to acquire. The new software development activities using CASE and 4GL required the redefinition of the software development process to use these tools. The new reengineering activities required the definition of new tasks for using tools to understand and restructure the legacy code. While the activities and tasks associated with software development were not always new conceptually, some were radically changed to take advantage of new tools available on the new infrastructure.

The changes in the information systems application domain tended to decentralize this domain. Many of the staff members moved to the separate operating units and became more concerned with finding the right tools to support local processes rather than maintaining consistent tool usage. For example, an operating unit may use any of five databases rather than the specific one previously available on the mainframe. The enterprise actively encouraged a move from software development to business domain analysis within the information systems application domain.

The enterprise strategy actually required two types of changes within the information systems application domain: the change in the environment infrastructure common across the enterprise

and a changes to the process of developing, acquiring and maintaining software. Changes in the environment infrastructure changed the range and quality of tools available to support the new and redefined tasks. Simultaneously, the redefinition of application domain process changed the need for specific activities and tasks. These two sets of changes are complementary in this strategy for this organization.

2.3 Stage 2: Financial Application Domain

With the implementation of the strategy in one application domain, the organization is in an unstable state. At this stage, the environment used internally for the information systems application domain is different from the mainframe and personal computer use that the information systems application domain must continue to support in other domains. The strategy called for migrating applications in domains that were not mission critical first. The enterprise chose to follow changes in the information system application domain with changes to the financial application domain.

The strategy called for migration to COTS tools and applications on UNIX servers and X terminals. Because of the prevalence of COTS relational databases rather than the hierarchical databases found in the enterprise's existing custom software, the enterprise's financial data had to be reengineered for the new databases. In addition, the new relational database tools required a reengineering of the tasks associated with manipulating this data; however, the new tasks accomplished generally the same goals using only slightly different processes and very different tools.

The changes brought about by the introduction of new tools could end at the activity level of this domain. The goals of the financial application domain and its major activities remained essentially unaffected by a change in the low–level tools in the short term. In the long term, the new environment infrastructure permits the financial application domain to select among a broader range of tools supporting more features allowing for more substantial process changes; however, these changes were longer term and outside the scope of the immediate changes brought about by the environment infrastructure change.

2.4 Stage 3: Changes to Mission Critical Areas

In the final stage of implementing the strategy, the enterprise applied the strategy to mission critical areas, especially the manufacturing and sales application domains. The changes brought about in these domains were analogous to the changes in the financial application domain. The changes were primarily to tasks through the introduction of new tools with only limited changes to the activities and application domains which utilize the tasks. The most significant change was introduction of the long term option of getting new, powerful tools with a wider range of features; however, the goals of the mission critical application domains were not changed limiting the effects of low–level changes to tasks. Activities fulfilling the application domain processes were still needed but these activities were slightly different in that they use new tasks using the new tools to complete the tasks.

An example of an affected activity crossing the manufacturing and sales domains was the ordering of custom configurations of telecommunications infrastructure products. This activity required the interaction of manufacturing and sales staff. The sales staff would take an order for a particular configuration, pass it to the manufacturing staff for validation, manufacturinig would then return

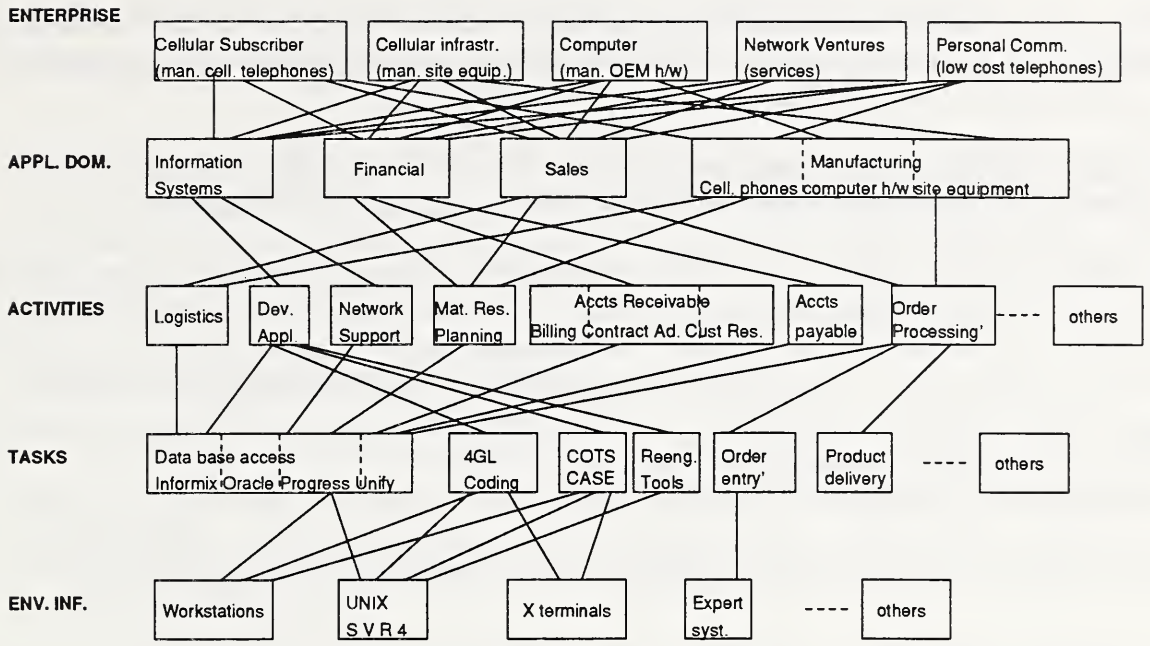


Figure 2.3: Final configuration

it to sales for additional work with the customer to correct the order. Finally, the sales staff would return a buildable configuration to manufacturing for construction. This process typically took 14 days. This activity was part of both an ordering process for customized equipment and the start of a manufacturing process (i.e., requirements analysis) for customized equipment. For this example activity, moving from mainframes to UNIX workstations, and from traditional software development to CASE tools, allowed for the development of a specialized expert system. The sales staff could use the expert system to check custom configurations, immediately reducing order time to one day. This is process reengineering at the task and activity level brought about by a technological change in the infrastructure.

Figure 2.3 illustrates the changes to the enterprise from implementing the strategy.

2.5 Observations on example

The enterprise has changed to accomodate the external stimuli on the organization, as given earlier. These changes mitigate the effects of some of these stimuli while new simuli affect the enterprise. The following summarizes the revised state of the organization, which will cause further changes in the future:

Market Forces: COTS software offers a wide range of options at a low price. Moreover, enterprise products are now commodities with prices for enterprise products continuing to decline.

Modeling Techniques: Enterprise uses decentralized systems administration. But now programmers and developers are retrained as business analysts. Information systems personnel use a rapid development model heavily automated with CASE tools and do no coding.

Metrics: Goal of software costing 1% of company revenue was achieved. Now maintenance of information system hardware and software requires a much reduced staff and software development requires fewer resources.

Technology Change: Continuing change in the marketplace in the availability of new, more powerful tools requires constant continuing monitoring of the marketplace.

The primary stimuli continuing on the company are economic: pressure on the pricing of its product (i.e., reduced income) and improvements in hardware and software tools offering better functionality at lower prices (i.e., reduced development costs). The enterprise must continue to examine the market and to respond to the availabity of better tools with more features (market forces and technology) by furthering the successful reengineering efforts which have already changed the companies models and measures of productivity.

This example illustrates how an enterprise can respond to the changing stimuli acting on it and take advantage of the available technology through a reengineeing effort. Change is not a single event but a process of stages, whose consequences the ITEM model can help focus on:

- Changes in infrastructure without process changes are initially self-limiting.
- Infrastructure changes can have repercussions long after the initial changes.
- Substantial change only happens when process and infrastructure changes are complimentary.

Infrastructure changes to automate an existing process or improve the tools available for an existing process will not, at least initially, profoundly change the enterprise. Such changes tend to only change the tasks or the activities of the enterprise, not its application domains or its goals and policies. In an abstract view, the enterprise is not seriously changed. Hammer [9] discusses many of the dangers of a reengineering effort limited only to infrastructure.

Substantial change in an enterprise has to include both infrastructure and process (i.e., how the enterprise uses that infrastructure) changes. For example, the changes in information systems were substantial affecting the models that information systems used to define its role. The process changes required by the new model of decentralized information systems services with personnel concentrating on business analysis, not lines of code, was possible because the infrastructure was available to make this new process work. Hammer refers to a true reengineering effort of this type as radical restructuring combining a new process and infrastructure.

The ITEM model can also be useful for describing the more subtle changes brought about from a deliberate infrastructure change with opportunistic process changes. In the example, the new infrastructure permitted other application domains besides information systems to gain access to new tools with new features. These new tools allowed personnel to perform tasks that had not previously been considered or to radically change old tasks. For example, the new workstations and expert system tools changed the task of ordering custom configurations of telecommunications products. This change had a substantial impact on the ordering process, but this change was not a primary reason for the information systems reengineering effort. Instead, it was a by-product of that effort as application domains use the new infrastructure with its new opportunities. Since these changes depend on tool availability in a variety of fields which is hard to predict, the ITEM model provides a way of describing the type of changes involved and how substantial they are for the enterprise.

3. Software Development Using the ITEM Model

Abstract

In this chapter the ITEM model is used to describe an application domain, in this case as applied to software engineering activities within an organization. The processes and automation components of the enterprise are described from the infrastructure up to the enterprise level of the model.

3.1 Software Engineering Environments

As given in Chapter 1, the ITEM Model describes the information technology components of an enterprise at five levels. In this section the model is applied to software development activities within an enterprise.

Currently, most discussions of information technology used in such organization emphasize only one of the following perspectives:

1. *Improved automation support:* Environments supporting interoperable software tools procured from various sources are an important factor in improving effective use of computer technology. Considerable effort has focused on developing better ways of specifying the computer systems that a heterogeneous group of vendors can provide. This includes the hardware platform, software environment infrastructure, and tools for effectively using the enterprise's information.

2. *Improved processes:* The services implemented in information systems provide only part of the answer for increasing productivity. How one uses those resources – the process model that drives an enterprise's use of information – is often as important, if not more so, than the underlying computer resources.

Both concepts – the environment and the process – are important to effectively use information technology. Therefore, any information model should integrate both aspects of the enterprise.

A software development enterprise develops processes using automation support at the five basic levels of the ITEM model. In the next subsection, one possible set of services that apply to each level of the model are given as an example of how the model can be used in specific domains.

3.1.1 Environment Infrastructure

At the base of the model is the automated computer system providing a core set of services for the automation of components of the adopted task, activity, application domain and enterprise processes. The NIST/ECMA SEE Framework Reference Model [17] and systems like UNIX, POSIX, Microsoft Windows, etc. describe models and products addressing information system infrastructure.

The infrastructure of information systems is undergoing a significant degree of standardization today. The frameworks reference model develoepd jointly by NIST and ECMA [17] is typical of today's approach towards defining the software component of an environment infrastructure. In the NIST/ECMA frameworks reference model (the *framework RM*), an environment infrastructure

or framework consists of 7 sets of basic framework services: (1) *Object management* services with data repository functions, (2) *Communication* services for passing information among environment components, (3) *Process management* services for building and using process models in the environment, (4) *User interface* services for permitting the user to communicate with components executing in the environment, (5) *Policy enforcement* services for instituting security and integrity constraints in the environment, (6) *Framework administration* services for maintaining the environment and tailoring it for individual use, and (7) *Operating system* services for implementing primitive functions that communicate with the underlying hardware platform.

The model is a service-based model in which each service describes an interface that supports some needed functionality. However, the details of this functionality (i.e., its signature of input and output objects) is not specified. Edition 3 of the framework RM lists 66 such services grouped into these 7 categories. It is not an architecture since it does not constrain the system designer in implementing each service and in combining the services into components.

The framework RM is not a requirements document for an environment. Instead, it is a catalog of potential services; each framework implementation can be measured against the framework RM to understand which services are present and which are not. It is unlikely that any environment will implement all the services. For example, there may be several methods for communicating information among the processes in the environment – storing information in the object repository, using a message passing communications service, using a remote procedure call communications service, or sharing common data storage in memory. It is unlikely that any specific tool set will need all of these options. Security (i.e., Policy Enforcement Services of the framework RM) is an additional example where enforcement of security issues throughout an environment may seriously impact performance if such enforcement is not warranted (e.g., a stand-alone computer operating in a home office environment).

Many of today's software interface standards define services relative to the services in the framework RM. Standards and proposed standards like X-Windows and Motif provide the interface for the user services of the framework RM, several of the POSIX standards (e.g., 1003.1) and X/Open's Spec 1170 define interfaces for operating system services, ECMA's Portable Common Tool Environment (PCTE) provides many of the object management services, and Sun's Tooltalk, and HP's Broadcast Message Server (BMS) provide many of the communication functions needed in a distributed system.

There are products – whether standards-based or not – providing components of the environment infrastructure. Systems like IBM's OS/2, SUN's Solaris, IDE's Software through Pictures, Cadre's Teamwork, Microsoft's Windows NT, are all attempts at solving the environment infrastructure problem by defining the set of application program interfaces (APIs) needed to support one facet of tool interoperability. Given a standard set of APIs, various applications can more easily interoperate on the same environment platform with a relatively high degree of common look and feel and internal data consistency.

While many approaches to an environment infrastructure have been implemented or are under development, not all relevant problems have been solved. Several key issues still remain:

1. The foremost problem is effectively *integrating* the tools of an environment. This, described shortly, is the primary functionality needed for appropriate automation of the Activity level that will allow personnel to use collections of software products to solve a single application problem.

2. The framework RM provides the set of infrastructure services needed to support the tools for many of an enterprise's tasks. However, is the set of services provided by this document sufficient? For example, is "weather forecasting" an appropriate service for a software development environment? (E.g., Is predicting whether your employees will be able to show up for work during a storm an appropriate software engineering development capability? It certainly would be for a personnel application domain.) This is unclear.

3. The framework RM was developed within a certain context around 1990. As such it has certain biases and limitations, among which are:

   - The framework RM was originally developed around a repository-based structure. As such, the set of communication services was originally very limited and while subsequently expanded, still needs further development.
   - The repository design of the framework RM did not originally include object oriented technology which evolved since the development of this model; additional service definitions to accomodate this technology may be needed.

4. The definition of other tools for specifying environment infrastructure is needed. These would include profiles of infrastructure standards, and mechanisms for specifying the integration methods supported by the tools.

3.1.2 Tasks

Tasks represent single steps towards the completion of the defined application domain process. Each development activity is composed of a set of tasks for performing that activity. Tasks usually are, but are not required to be, automated or supported with specific automated tools that transforms inputs into outputs. For example, the activity of building software within the waterfall life cycle [20] application domain process requires a coding task which may be performed manually and supported with a text editor or automated with a code generator. Combinations of tools or tool sets can provide assistance in the performance of individual tasks and activities.

The PSESWG Project Support Environment Reference Model [6] (called PSE RM) developed the concept of *end-user services* to describe these software products. The end-user services suggest a partial list of tasks for a software development enterprise.

For example, the PSE RM developed end-user services in four different application domains – Technical Engineering, Technical Management, Project Management, and Support services. Each of these, in turn, is divided into the set of services needed to support individuals engaged in specific activities or tasks. For example, the Technical Engineering services include System Engineering services, Software Engineering services, and Process Engineering services. The Software Engineering services include the more familiar tasks of requirements definition, design, coding, static analysis, testing, etc. Typically, a single product implements one of these fine-grain services. There are products available supporting or automating most of these services. For example, within the Software Engineering application domain, the following sample tools exist:

Service	Software tool
Software Design	MarkV ObjectMaker
Software Simulation	Arcadia project's Chiron GUI builder
Software Generation	UNIX's YACC
Compilation	Compilers for C, FORTRAN, Ada, ...
Software Static Analysis	NASA's Static Analysis Program (SAP)

The application domain process model selected by the enterprise defines activities and tasks, the processes used to implement them, the services used to support the set of them, the tools used to implement them and the relationship among those tools. The application domain and the activity level process defines how the end-user services and the tools implementing those services should be related in an environment. For example, PSE software engineering services include both a software verification service and a software testing service. How one implements each service and connects the implementations are decisions made locally to tailor the environment to support the enterprise's defined process.

While the PSE RM has defined a candidate set of end-user services for the systems and software engineering application domains, outside of the software development world, little progress has been made to develop catalogs of necessary end-user services in other domains.

There are few standards or standards activities related to defining tasks and end-user services. Programming language syntax and semantics have been standardized for most common languages (e.g., C, FORTRAN, Ada) and many data base languages do have standard interfaces (e.g., SQL). There are also industry "benchmarks" for testing some systems and standards like POSIX 1003.1 and several language standards do have test suites for testing conformance. ISO/IEC/JTC1/SC7 has defined software life cycle processes and begun defining the tasks that make up these processes in areas like configuration management. Outside of these activities, there have been few efforts to define compatible interfaces for end-user services.

3.1.3 Activities

In software development, software engineers must engineer software (e.g., specify and design it), build software (e.g., code and test it), certify software (e.g., verify, validate, perform quality assurance on it), and distribute software (e.g., enhance and maintain it). Each of these activities supports a portion of the application domain. Each organization tailors its software development process into activities which address specific concerns in its own application domain model. Various process models are available for modeling activities, such as object oriented design or cleanroom software development, while others models handle configuration management, quality assurance, and specific development tasks like software coding, testing, etc.

Activities are defined sequences of tasks, supported by end-user services, which have a clear initiation point, progression and end point and have the goal of producing a product. This does not rule out iteration among tasks until conditions are met. The definition of an activity includes policies on when and how the activity can be initiated (preconditions on the activity), the circumstances of moving from one task to another (pre- and postconditions on each task) and how the activity can be concluded (postconditions on the activity).

The PSE RM defined activities as sequences of user actions supported by sets of defined services as opposed to tasks which are user actions supported by a single service. This mapping of activity to multiple services implies the need for multiple tools to complete that activity. For example,

the PSE RM defined software quality assurance (SQA) as an activity that utilized the metrics, verification, testing and configuration management end-user services of the existing PSE RM. SQA was not deemed to be a separate service (e.g., implemented as a single computer tool), but instead was a process composed as the enactment of several existing services.

The concept of tool integration, mentioned above under environment infrastructure, is crucial to effectively providing automated support for activities. In order to automate activities with a tool set, there must be a seamless way to pass information and control among the tools. For example, the build software activity mentioned above generally uses the following tasks and tools:

- Use a **design compiler** to aid in the translation of design to source program text.
- Use a reuse library **browser** to identify any available components to reuse.
- Use an **editor** to enter source program text.
- Use a **compiler** to translate the text.
- Use a **linker** to assemble the separate program components into an executable version.
- Use a **static analyzer** to analyze the structure of the source program.
- Use a **metrics evaluator** for determining the complexity of the source program.
- Use a **verifier** to prove the correctness of the source program.
- Use a **source code debugger** to monitor program execution.
- Use a **testing tool** to execute and test the source program.
- Use a **configuration management tool** for maintaining libraries of source programs.

Two immediate observations from this list of tasks:

1. Today, although all of these tools would be useful, they cannot be built into a single software engineering environment economically. However, this set of tasks is necessary (whether computer-based or manual) for most software development today, and the proposed model in this paper must be able to describe an environment containing all these tools and processes that would use the full tool set.

2. Information must be able to be passed among the set of listed tools: **editor, browser, design compiler, compiler, linker, static analyzer, debugger, metrics evaluator, verifier, testing tool** and **configuration management tool**. This is the integration problem. While individual tools are relatively easy to build, accepted techniques for transferring data and control among tools need to be further studied and agreed upon mechanisms need to be developed.

There are a few standards, or standards activities, in this area. Those available tend to focus on software development or other types of product development. Standards like Microsoft's Object Linking and Embedding (OLE), Common Data Interchange Format (CDIF), Standard for the exchange of product model data (STEP) [23], Electronic Design Interchange Format (EDIF) [12], Common Object Request Broker Architecture (CORBA) and PCTE are trying to address how heterogeneous tool sets may interoperate. Standards like DIS 12207-1 Software Life Cycle Processes define processes and activities in such areas as software development, acquisition and configuration management.

3.1.4 Application Domains

If software development is important to the enterprise, what is the process for building software? Generic processes exist for addressing some of these goals: DOD-STD-2167A for defense system software development [8] and the spiral model of software development [5] are possible choices. There are also processes like the SEI CMM [18] for measuring and improving the development process. Measurements of the effectiveness of application domain processes to meet the enterprise level goals begin to address the "engineering" of information technology. Effective automated support for processes requires extensive tool interaction and customization in software environments.

Application domains are focus areas of the enterprise that result in identifiable products. However, "product" does not necessarily mean items sold by the enterprise. Products may be internally used, such as production of the payroll "product." An application domain process will consist of a sequence of activities with the identifiable product as a goal. Application domains can produce products which are feedback for redefining tasks and activities and for reentering the process. Application domain models define the mechanisms an organization uses to build its sets of activities and the organization's constraints on these activities. These activity definitions determine the organization's need for end-user services. The task interconnections in the activity imply the need for corresponding service interconnections implemented in some environment. Much of the current interest in process modeling addresses the set of processes needed to solve problems in some application domain. Process model activities in the software development domain include:

- *Defining the enterprise's Software Development models.* The process of developing software is certainly a major concern, and development models have been under study for many years. Most organizations use the so-called "waterfall model" and processes similar to MIL-STD-2167-A standards for developing software. In this model, each phase of the development process must be completed before the next phase can begin. The process progresses from requirements to specification to design to code to test for the complete software system. The model is *product-based* since moving from one activity to the next generally requires the completion of a milestone product – for example, a design document before coding begins, source programs successfully completing unit test before integration testing, an integrated system passing integration tests before beginning software quality assurance validation, etc.

  However, variants to the waterfall model are starting to appear. Boehm has proposed a "spiral model" that emphasizes risk reduction and prototyping as drivers even though the ultimate development follows a waterfall-like process. Rather than view the development process as a progression of deliverable documents, development is viewed as successive prototypes needed to reduce the overall risk of building a product. The cleanroom process uses similar phases to the waterfall model; however, cleanroom places a greater emphasis in design verification with a corresponding lessening (if all goes well) of testing during the coding phase.

- *Quality improvement.* An important consideration is the improvement in the quality of software by improving the quality of the software development process. The assumption behind improving the development process is that improvement in the process will lead to improvement in the product. Two well-known examples of quality improvement processes follow:

  (1) The SEI Capability Maturity Model (CMM) is a process an organization can use to improve its software development activities. The CMM defines a process for an organization

to follow to investigate its own development processes, to institute management controls and measurement guidelines and to improve its understanding of how it does its business.

(2) The NASA/GSFC Experience Factory [4] is a model that grew out of NASA-Goddard's Software Engineering Laboratory (SEL). This is a bottom-up approach where an organization tailors an existing development process for process improvement. Its ultimate goal, however, is similar to the CMM in getting an organization better prepared to develop software.

There is additional work in developing notations for designing development processes. Systems like Marvel, Process Weaver, and others, are all attempts to provide automated support for a defined process. therefore, encouraging its use.

3.1.5 Enterprise Models

An enterprise consists of the continuing processes which define the functioning of the application domains that produce the enterprise's products and services. Within the software development area understanding the relationship of software to the organization's goals is a first step. Real-time embedded applications (e.g., computer controls in an automobile) reflect a different set of issues than a pure software development (e.g., producing a new spreadsheet program for use in a desktop workstation). Using methods such as sequential or "pipeline" processes are major enterprise-level decisions. Alternatives may consist of parallel or "concurrent" engineering approaches [13]. How one implements such processes depends upon the specific application domain that the enterprise is concerned about.

The enterprise model also defines the policies used in executing the processes, the relationships among these processes and the starting and termination of processes. Risk analysis is a major concern of business policy-making. This level is concerned with the policies of the enterprise's business plan. How does an organization define its development processes? Is the organization often involved in building many new and different products where the risk of an incorrect design extremely high or is it involved in repeated development in similar applications? In the former case a process like the spiral model will minimize the exposure to cost risks by forcing successive prototypes to deal with the unknowns of the development process. On the other hand, for the latter case, a more standard waterfall process may be less expensive since there is less risk of a faulty design.

Does a CMM or an Experience Factory process improvement approach fit better with other processes in the enterprise and with the enterprise goals? What are the risks in developing either model. Little has been done to develop a software environment supporting enterprise level processes, process management, and decision making.

3.2 Sample use of Model

Consider the following hypothetical problem: A software company needs to develop a new software tool – one very different from products previously developed. The ITEM model can specify the various decisions the organization must make according to the relevant model levels. Each level may specify behavior (i.e., processes) or tools (i.e., automation) for carrying out the necessary steps at that level.

Enterprise. The application domain will be software development. There is considerable risk since similar products were not developed previously by the organization.

Application domain. Since software development is the goal, and since the risk is high, a spiral development methodology will be used. The CMM will be used to evaluate activities in order to understand how development proceeds, to collect data on the development, and to improve the process in the future. Systems like Amadeus [21] or NASA's SME [24] may provide for some of the measurement processes needed.

Activities. Spiral development requires an iterative development process. Activities could be: (1) Develop requirements; (2) Prototype user interface; (3) Prototype data repository; (4) Develop first release. Each of these separate activities have stages of: (a) Design; (b) Develop; (c) Evaluate current risk. Process improvement requires modifications to these activities to plan for data collection. Tools such as Process Weaver or Marvel [1] can be used to automate the process of task scheduling in complex developments.

Tasks. For each of the activities previously given, it is necessary to break them down into specific work items. For example, prototype user interface could be: (1) Develop design using MarkV Objectmaker; (2) Develop prototype of interface with Arcadia Chiron tool; (3) Compile prototype with Ada compiler; and (4) Edit text with emacs editor. In each case there is a tool associated with each task and the set of tasks combine to form an activity.

Infrastructure. There is the obvious requirement that the above tasks must be executed. An environment framework, such as a SUN workstation executing UNIX and POSIX 1003.1 compliant with X Windows and Motif user interfaces, can be specified. Communication may be enabled via SUN's ToolTalk. Other environment infrastructure products may also be specified.

In reality, for many organizations, the infrastructure is already in place and the set of tasks must be developed to be compatible with this infrastructure.

Today pieces of the ITEM model are understood. Work on the framework RM and PSE RM provide a good baseline for automation issues at the implemented environment level. Work on process modeling provide the basis for determining some of the higher-level attributes - but there are many gaps (e.g., what are the various models at each level? How is measurement applied across all levels? What standards and products can be used to implement services at each level?) Current efforts are to determine how to extend the model and fill in these gaps.

4. Evolution towards a National Information Infrastructure

Abstract

This chapter describes some of the structure that the NII may acquire and how various end user information services, brokers of those services and providers of NII capabilities may be interrelated.

4.1 Engineering the NII

The previous section applied the ITEM model to describe the reengineering of an organization. This is a microcosm of the changes that will affect the IT industry as the NII evolves. All processes and automation will adapt to the technology available through the NII; conversely, the telecommunications, broadcast, and information technology industries will evolve to create NII services in response to industry and government needs.

A NII will be a "virtual enterprise" composed of different suppliers, vendors, agents, brokers and users. Today, there are vertical markets for providing information infrastructure services in a series of separate enterprises (see Fig. 4.1): The local telephone company provides the wires to your home and implements voice services such as call initiation, call forwarding, voice mail, and call waiting with connections to several long distance carriers; the local cable franchise receives signals from a broadcasting satellite, and through cable entering your home, provides video services; an information technology service provides electronic mail and bulletin board services for its customers via connections through the local telephone service. Although there are several companies providing each form of service, some (e.g., local telephone service and cable television) have a monopoly on providing that service within a given geographical area. It might surprize most to realize that many consumers today have the ability to receive five distinct data streams in their homes:

- telephone service (via "twisted wire" pairs entering home)
- cable television (via cable entering home)
- two-way cellular telephone (via radio transmission)
- broadcast service (via traditional radio and television)
- broadcast satellite television (via home satellite dish)

Much like the changes in the telecommunications arena in the mid-1980s which resulted in multiple long distance carriers for communication services, in the future there may be multiple providers of basic services. Rather than contracting with "the telephone company" or "the cable company" in their area, consumers may be able to obtain digital services from one of many "bitway" providers. Similarly, there may be multiple suppliers of value-added services. There may be several cable television franchises, each having their own assortment of cable channels purchased from "video wholesalers." The cable television franchisee takes on the role of broker in permitting a user to purchase one package of cable channels delivered to a home over one of several bitways. This provides multiple ways (read that "competition") for which such services can be made available.

Consider the following hypothetical situation (Fig. 4.2). A user at home (second from left) wishes to contract for cable channels 1 and 4. Both brokers 1 and 2 have packaged those services.

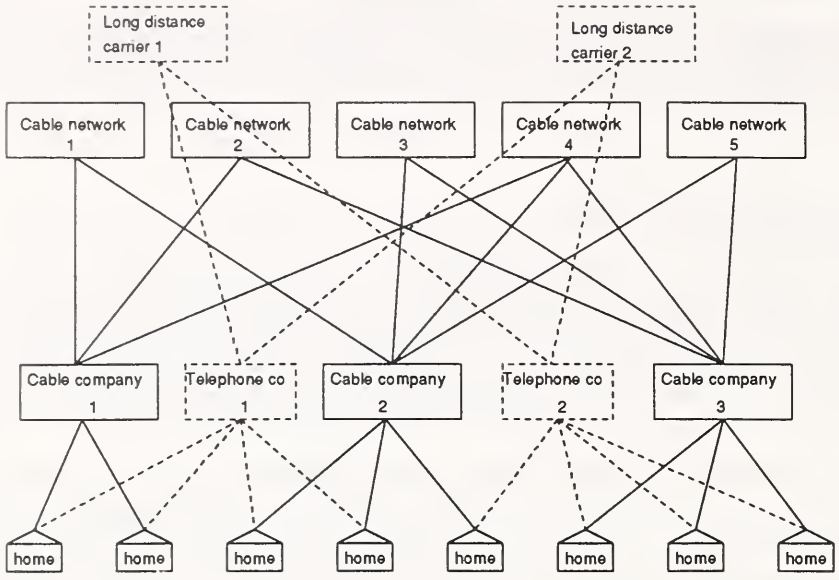


Figure 4.1: "On ramps" to information superhighway

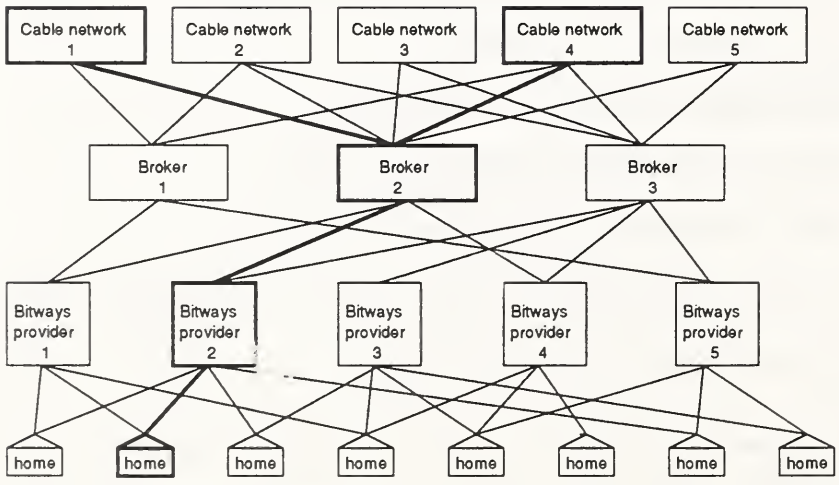


Figure 4.2: Possible service interconnections

Broker 1 makes its products available via bitways providers 1 and 5 while broker 2 provides communication services via bitways provider 1, 2 and 4. The end user, who has bitways provider services available from both provider 1 and 2, can choose to obtain cable services from broker 1 via bitways provider 1 or from broker 2 via either bitways providers 1 or 2. The figure shows one of these 3 possible paths. It should seem clear that this example can easily translate into other industries besides providing cable television access.

What is the evolution towards Fig. 4.2 and what are the economic implications of this figure? Much like the example described in Chapter 2, today's model of Fig. 4.1 will evolve to something like Fig. 4.2. Initially, many such alternate paths could develop and the evolution of existing vertical markets among many industries need to be evaluated with, for example, the ITEM model. The set of infrastructure services provided by the bitways must be identified, as well as the services provided by the mid-level brokers and the services addressed by the application domains providers. Understanding consumer demand will permit the NII to evolve and allow components of the NII "virtual enterprise" to meet those demands.

Eventually, as this technology matures, the industries will probably coalesce into a few dominant players – assuming antitrust and monopolistic business issues are addressed adequately. Understanding the enterprise evolution of activities, tasks and infrastructure services among brokers, providers, and suppliers may all be made more manageable with a model such as proposed here.

4.2 Conclusions

This paper described the outline of the Information Technology Engineering and Measurement Model – to understand how information systems are used in an organization. Understanding this as information technology evolves provides a mechanism of change as organizations evolve over the next decade. It is best to develop a model that addresses both the processes needed to support an enterprise and the set of computer resources that are able to carry out as much of this process as possible.

This model is one approach towards describing the technology needed for the National Information Infrastructure, and this paper presented two examples of the use of this model – to describe a software engineering development environment and an enterprise-wide information structure.

5. References

[1] Barghouti N. S. Supporting Cooperation in the Marvel Process-Centered SDE. *Proc. of the ACM SIGSOFT Fifth Symposium on Software Development Environments*, McLean VA, *ACM SIGSoft Software Engineering Notes* 17(5):21-31, December 1992.

[2] Bartlett J., Cost crunch forces IS downsizing, 14-19, *Uniforum Monthly*, June, 1994.

[3] Basili V. R. and H.D. Rombach, The TAME Project: Towards Improvement-Oriented Software Environments, *IEEE Trans. on Software Engineering* (14)10:758-773, 1988.

[4] Basili V. R., G. Caldiera and G. Cantone, A Reference Architecture for the Component Factory, *ACM Trans. on Software Engineering and Methodology*, (1)1:53-80, 1992.

[5] Boehm B., A spiral model of software development and enhancement, *IEEE Computer* (21)5:61-72, May, 1988.

[6] Brown A., D. Carney, P. Oberndorf, and M. Zelkowitz (Eds.), *Reference Model for Project Support Environments*, Version 2, NIST Special Publication 500-213, October, 1993.

[7] *Corporate Information Management: Process Improvement Methodology For DoD Functional Managers,* Second edition, D. Appleton Co., 1993.

[8] Dept. of Defense, Military Standard: Defense System Software Development, DOD-STD-2167A, 29 February 1988.

[9] Hammer M. and J. Champy, *Reengineering the Corporation*, HarperCollins Publisher, New York, 1993.

[10] IEEE, Draft Guide to POSIX: An Open Systems Environment, P1003.0, D16, August, 1993.

[11] Joos R., Software reuse at Motorola, *IEEE Software* (11)4:42-47, September, 1994.

[12] Kahn H. and R. Goldman, The Electronic Design Interchange Format (EDIF): Present and Future, 29^{th} *ACM/IEEE Design Automation Conference*, 1992, 666-671.

[13] Malone T., K. Crowston, J. Lee, and B. Pentland, Tools for inventing organizations: Toward a handbook of organizational processes, *Proc. of* 2^{nd} *IEEE Workshop on Enabling Technologies Infrastructure for Collaborative Enterprises*, 1993, 72-82.

[14] NIST, Integration definition for function modeling (IDEF0), FIPS Pub 183, NIST, December, 1993.

[15] NIST, Information Infrastructure Task Force, What It Takes To Make It Happen: Key Issues For Applications Of The National Information Infrastructure, January 25, 1994.

[16] NIST, Information Infrastructure Task Force, Putting the Information Infrastructure to work, NIST Special Publication 857, May, 1994.

[17] NIST, *Reference Model for Frameworks of Software Engineering Environments*, Edition 3, NIST Special Publication 500-211, August, 1993.

[18] Paulk M. C., B. Curtis, M. B. Chrissis and C. V. Weber, Capability Maturity Model for Software, Version 1.1, *IEEE Software*, (10)4:18-27, July, 1993.

[19] R&D for the NII: Technical Challenges, Symposium, February 28 - March 1, 1994, EDUCOM, 1112 16 St NW, Washington, DC 20036.

[20] Royce W. W., Managing the development of large software systems: Concepts and techniques, Proceedings Wescon, IEEE Computer Society, 1970.

[21] Selby R., A. Porter, D. Schmidt, and J. Berney, Metric-Driven Analysis and Feedback Systems for Enabling Empirically Guided Software Development, ACM/IEEE 13^{th} International Conference on Software Engineering, Austin, TX, May, 1991, 288-298.

[22] Technology for Economic Growth: President's Progress Report, White House, Washington, DC, November, 1993.

[23] Trapp G., The emerging Step standard for product model data exchange, *IEEE Computer*, (26)2:85-87, February, 1993.

[24] Valett, J. D. Automated Support for Experience-Based Software Management, *Proceedings of the Second Irvine Software Symposium (ISS '92)*, Irvine, CA, March 1992.

[25] Work B. and A. Balmforth, Using abstractions to build standardized components for enterprise models, *Proc. 1993 IEEE Computer Society Software Engineering Standards Symp.*, 1993, 154-162.

A. Acronyms and Abbreviations

API	Application Program Interface
ATM	Asynchronous Transfer Mode
BMS	Broadcast Message Server
BPS	Bits per second
CASE	Computer Assisted Software Engineering
CIM	Coporate Information Management
CMM	(Software Engineering Institute's) Capability Maturity Model
COTS	Commercial Off The Shelf (software)
ECMA	European Computer Manufacturers Association
EDI	Electronic Data Interchange
EDIF	Electronic Design Interchange Format
EFT	Electronic Funds Transfer
GSFC	(NASA) Goddard Space Flight Center
GUI	Graphical User Interface
IDEF0	Graphical activity modeling language
ITEM Model	Information Technology Engineering and Measurement Model
MBONE	Multicast Backbone
MIS	Management Information Systems
MPEG	Motion Picture Experts Group
NASA	National Aeronautics and Space Administration
NII	National Information Infrastructure
NIST	National Institute of Standards and Technology
OLE	(Microsoft's) Object Linking and Embedding
PC	Personal Computers
PCTE	Portable Common Tool Environment
POSIX	Portable Operating System for Information Interchange
PSE	Project Support Environment
PSESWG	(U. S. Navy) Project Support Environment Standards Working Group
QIP	Quality Improvement Paradigm
RM	Reference Model
SEE	Software Engineering Environment
SEI	Software Engineering Institute
SEL	(NASA/GSFC) Software Engineering Laboratory
STEP	Standard for the exchange of product model data
SQA	Software Quality Assurance
VCR	Videocassette Recorder
YACC	Yet Another Compiler Compiler