Designing Web Information Systems for Internet Commerce through the Virtual Organization Model

Kam Hou VAT
Faculty of Science & Technology
University of Macau, Macau
Fax: (853) 838-314
fstkhv@umac.mo

Abstract

This paper investigates the design of digital universities in the form of Web information systems (WISs) prototyped for activities including electronic commerce (EC). Specifically, we describe the business and technology architectures constituting our virtual university model (VU). We interpret the WISs as an iterative means to realize the services offered by the VU, and discuss the importance of designing an enterprise information architecture (EIA) and constructing a repository of reusable strategic assets (business and technology). Also we explain the ideas behind our methodology to transform a bricks-and-mortar university to its clicks-and-mortar form. These include integrating its EC vision and strategy, architecting business processes and technology applications to operationalize the strategy, and creating the architectural components for future reuse. The paper concludes by discussing the corresponding software architectural approaches required to develop the different WISs applications supporting the EC activities of the VU.

1. Introduction

The notion of virtual organization, according to Mowshowitz [19, 20], could be expressed as a set of principles for managing goal-oriented activity based on a categorical split between task requirements and their satisfaction [12, 18]. In this formulation, the virtual organization model can be conceived as an approach to management that explicitly recognizes the conceptual distinction between functional requirements and the means for their realization in practice, as well as providing a framework for accommodating dynamic changes in both requirements and available services. In a dynamically changing organizational and technological environment, it is essential that we can logically separate the requirements from the means for their satisfaction. That way, management could create an environment in which the means for reaching a goal are continually and routinely evaluated in relation to explicit criteria. Such management structure ensures that requirements are satisfied as appropriately as possible. It is believed that this idea could be adopted in a variety of settings to enhance the efficiency and effectiveness of the underlying information systems and to motivate the participants involved to reflect on organizational goals. We have chosen to experiment with this notion in the design of our digital university, referred to as the *Virtual University* (VU). And our VU involves the construction of numerous *Web-based information systems* (WISs) [15] for different functional requirements including those related to the activities of electronic commerce over the Internet, commonly called *Internet Commerce*.

2. The VU's Virtual Organization Model

The virtual organization model behind our VU, makes explicit the need for dedicated management activities (called meta-management) which identify the abstract requirements needed to realize certain objective while simultaneously but independently, investigating and specifying the concrete means for satisfying the abstract requirements. Formally, the VU can be characterized by a number of activities. These include [18, 19, 20]: formulation of abstract requirements; tracking and analysis of concrete satisfiers; dynamic assignment of concrete satisfiers to abstract requirements on the basis of explicit criteria; and exploration and analysis of the assignment criteria associated with organizational

goals. The VU exploits the advantages of switching through dynamic assignment of available services to requests according to explicit criteria. It is believed that the VU's ability to switch systematically based on explicit formulation of goals, allows for a high degree of flexibility and responsiveness in improving resource utilization, achieving better quality products, strengthening managerial control, and providing cost-effective services. Indeed, the VU, being an innovative form of organization, promotes reflection by providing a meta-management framework for exploring requirements, satisfiers, assignment methods and criteria, to assess and optimize the organizational performance.

3. The VU's Vision of Internet Commerce

With the rapid advances in networking technologies and the commercialization of the Internet, universities nowadays are well poised to deliver customized educational content worldwide for life-long learners [10]. Yet, this vision, which looks at education as an information industry, often requires establishing some electronic infrastructure within the physical university, in order to take advantage of the new technologies and the business opportunities. Our VU's electronic infrastructure is supported by the incremental creation of WISs, which are interpreted as an iterative means to realize the various services offered dynamically by the VU. In today's electronic markets, the VU, as supplier of education, will have to meet the specific needs of a diverse, globally distributed customer base. New players will certainly enter the education and training markets. The competitive business climate often requires a team of separate business partners working together to meet the diverse complicated market demands. High-performance computer networks allow companies to collaborate electronically, through data and technology sharing, to assemble the creative ideas to develop complex products, and to achieve production agility. This arrangement, often called an industrial *virtual enterprise* [11], is a consortium of independent member companies coming together based on cost-effectiveness and product uniqueness without regard for organization sizes, geographic locations, computing environments, technologies deployed, or processes implemented. Virtual enterprise companies share costs, skills, and core competencies that collectively enable them to access global markets with world-class solutions their members could not deliver individually. When applied to the VU context, the virtual enterprise could include partners in the form of other universities, research centers, major publishers, tools-vendors, or other education brokerages.

4. The VU's Digital Framework for WISs

To understand the VU's digital framework in support of WISs, we distinguish the information technology (IT) applications both inside and outside the system firewall. The former often termed intra-organizational applications, are internally focused to enhance the existing relationship between parties within the organization, typically by promoting the efficient exchange of information. The latter called inter-organizational applications, are externally focused to facilitate new business relationships and attract new customers via the organization's Web site. The VU's WISs focus on both the intra- and inter-organizational IT applications, and they are respectively deployed on the VU's Intranet (internal private network) [31], and Extranet [22], which currently has two connotations. The first represents an inter-organizational collaborative network using Internet technology to provide seamless communication services among member organizations to engage in cross-application information messaging. The goals are often to enhance efficiency and reduce time to market of business-to-business deliverables, and to increase the competitiveness of the entire consortium versus other virtual enterprise. The second denotes part of an organization's Intranet made accessible to other business partners or customers for such services as data mining and processing. They are essentially inter-organizational decision support systems where an external partner uses a Web browser to drill down and pull the desired information into the client applications.

5. The VU's Positioning of WISs

Our VU positions the WISs as an iterative means to realize the services offered dynamically according to the ongoing functional requirements of the business models. Technically, WISs, represent the important information systems (IS) efforts geared toward exploiting the benefits of the Web platform. They are the systems that organizations, their clients and business partners, use to conduct Intranet-based and Extranet-based distributed applications including Internet commerce. According to Isakowitz, et al. [15], unlike Web pages designed largely for leisure

browsing, WISs enable users to perform work, and is usually tightly integrated with other non-WISs such as databases and transaction processing systems. Our VU believes that in WIS development, user participation is as critical as it is for traditional IS development. And WISs should be developed by using the same disciplined system development principles, rigorous business value assessment, and user-centered approaches that are required to build successful non-WISs [7]. Meanwhile, we also believe that WISs development is sufficiently different from traditional IS development in that it requires new approaches to software engineering, because WISs have the potential to provide distributed computing environments among geographically dispersed coworkers. These differences include such Web development details as [28]: 1) navigation structure designed to support specific work flow; 2) structured data modeling representing relationships among pieces of information; 3) features that enable users to process business data interactively; 4) support for distributed collaboration work style; and 5) link referential integrity for mission-critical tasks. Ultimately, it is believed that WISs should help organizations to enhance their competitiveness and facilitate differentiation of their products and services from other competitors' through standards of high quality.

6. The VU's e-Transformation Methodology

The electronic transformation (e-Transformation) from a physical university to its digital counterpart, as in the case of our VU, requires an objective methodology. And this methodology must be instrumental to creating a productive and efficient electronic university model, which enables us to follow an iterative development sequence. This means being able to plan and prepare for a launch based on a new business model within a very short cycle time. In particular, this model should enable our VU to launch and learn, and then incorporate those lessons and launch again. Actually, this can be accomplished only if we have an agile operation based on a reusable business and technology infrastructure, and supported by a repository of reusable business and technology assets. We try to characterize the methodology to achieve this model as follows: First, we need to define an electronic vision (e-Vision) for our VU, to bring all of its real-world and virtual-world strengths together in a re-configurable constellation. Second is to define the VU's business architecture, encompassing its associated business models, processes, and applications which will let us move from vision to reality. Third, we have to entail a corresponding technology architecture that allows an iterative implementation of the business architecture. Fourth is to create a reusable infrastructure of both business models and technology applications, which allow us to recycle every piece of learning, time after time, and in as little time as possible. In other words, we recognize that becoming an electronic University is not simply a technology issue to be managed by an IT department. Instead, the e-Transformation itself involves business process engineering and re-engineering, and it is a core strategic issue, requiring meticulous planning before construction. It is about molding selected aspects of the running university into whatever the reengineered vision of the educational process and the market (global and local) demand that they be. And it is about setting long-term goals to refocus the business of education. Hence, it is about business as much as it is about technology, and as such it must be managed directly by a team of integration specialists (executives, administrators, professors, technologists) who can walk the line between university-wide (corporate) strategy and IT issues. Oftentimes, this requires IT leaders to learn business, and business leaders to learn technology. In today's complex business-to-customers (b-c) and business-to-business (b-b) environment, it is believed that the core of an electronic university is a backbone of inter-organizational business processes which cross organization boundaries to include external stakeholders. And a successful platform for the electronic applications provided by the WISs of our VU, must involve, not only customers and suppliers, but also internal processes, employees, and back-office functions, as well as external partners.

7. The VU's Business Architecture

The VU's business architecture comprises components of the university architecture that are directly related to modeling the business functionality of the organization. Its design should be guided by the overall re-engineered vision of the educational process, and must remain grounded in the realities of university execution, which include such tangible issues as product (program/course) development plans, and customer (student) relationship management. The VU's business architecture is generally divided into three distinct components [14]: e-Business models, e-Process

models, and e-Application models, where "e" denotes electronic.

- *e-Business Models*. These models provide a high-level perspective of the VU's business structure, which include determining the organization's target market and primary audience for its goods and services, the optimal product and service mix for each market segment, and the revenue streams that will be generated by the product-service-customer mix. They also include such elements as branding, distribution channel, partnership strategy, and the issue of ownership of intellectual property and physical assets. Moreover, the business models should include careful analysis of university resources to ensure that the organization reuses assets that provide value. More importantly, the models must define critical success factors, and the return-on-investment analysis and measurement criteria in order to create a balance scorecard to keep the university on track.

- *e-Process Models*. These models describe the internal and external processes that define the organization's day-to-day behavior. They must reflect the university's information strategy, and the individual business models chosen for implementation. The electronic process modeling typically involves defining and working with business entities such as customers, suppliers and departments, which are then assembled to form complete inter-organizational business processes, reusable in multiple iterations of business process engineering.

- *e-Application Models.* These models are aimed to represent the electronic applications that will be developed to streamline business processes. Specifically, they outline the overall application functionalities from the end-user perspective, in the form of a user-interface mock-up, which allows users to step through the process through the application's navigation aids. Often, the application functionalities are supported by some data and object models, which describe the underlying data structure and usage for a target application.

8. The VU's Technology Architecture

The VU's technology architecture is aimed to translate the organization's business vision into effective electronic applications that support the re-engineered intra- and inter-organizational business processes. And it is typically composed of distinct stages of development such as [14]: e-Application rules, e-Application data, and e-Application distribution, where "e" denotes electronic.

- *e-Application Rules*. Every business process has rules to govern the operation. And e-Application rules are the technical mechanisms, which enforce business rules. Typically, these rules are unique to each and every implementation. And they must be modeled to each university's policies and specifications. In terms of implementation, such rules are generally embedded within various software artifacts, which implement the necessary business requirements. When a transaction is entered into the computer system, the software artifacts interact to see if it is allowed to complete the transaction. That way, the application logic is made to mirror the business logic prescribed during the e-Process modeling phase of the business architecture development. Typical components of the e-Application rules include: business objects, and application frameworks, which involve different levels of technical design.

- *e-Application Data*. The e-Application Data comprises data stored and manipulated by the VU's applications through open standards such as JDBC or ODBC. Often, the e-Application Data simply consists of relational databases. However, some data management systems could include object databases that store data in encapsulated software objects rather than as entries in a database. No matter how data is stored, it must be monitored and distributed to applications within and outside the organization in a manner that first anticipates data requirements and second enforces data access policies to what is often sensitive business information.

- *e-Application Distribution*. Since the VU's platforms have to cross internal departmental boundaries and even bridge multiple member organizations of the virtual enterprises, they must by necessity work with a wide variety of existing technology implementations. The cornerstone of e-Application distribution is a distributed architecture, which allows application resources to be located on individual application servers. And these servers are connected by a network infrastructure, which provides a backbone of communication between the multiple distributed platforms of the VU's, and which

communicates using middleware standards such as CORBA and COM/DCOM.

9. The VU's Reusable Asset Repository

The VU's e-Transformation methodology requires iterations of references and modification of the components developed in the business and the technology architectures of the university. This requirement implies the importance of a reusable asset repository for storing various business-specific and technology-related components. The VU's business repository stores information, which we can use to standardize definitions for business and process models. Typically, users can archive existing process components, including business entities such as purchaser and supplier, activities such as approval and requisition, and processes such as procurement and approvals. These archived entities can then be recalled later by coworkers in other departments to be reused or modified for new process models. Similarly, the technology repository stores technology resources such as business objects, pre-built and purchased components, developer documentation, application design parameters and other technology standards. Actually, developing repositories are organizational steps taken to capture and store the tacit knowledge of people throughout the organization and to provide services to business partners across the organizational boundaries. So, repositories are essential elements of the organization's knowledge management platform. It is believed that information repositories should be codified according to the final business use supported by the information. To enable this specific segmentation, we should first focus on a single particular area of application, called a domain, and collect only components which are known to be useful in that domain, then expand the required repository domains as the range of reusable assets grows. We should also be concerned with the ownership issue as to who will own, manage and maintain the information in the organization. The end result of a successful business and technology repository should be a reusable asset that enable the VU to share information to reduce its application costs, and the cycle-time for new versions of applications and process models. More importantly, it should provide strategic advantage in the form of faster time-to-market and responsiveness to customer needs, which is especially important in life-long learning where the needs of the students evolve with their careers.

10. An Example of Enterprise Modeling for the VU

In this section, we shall start from a typical business-to-customer (b-c) electronic commerce (EC) scenario for the VU, and then describe the necessary business and technology components to be deposited into the reusable asset repository. These components will become the constituents to be used in the development of an appropriate WIS including the various distributed applications, to fulfill the e-Vision behind the scenario.

10.1 VU's e-Vision for B-C EC Scenario

The VU's e-Vision looks at education as a business and the potential students as customers. A typical business transaction starts when customers, on approaching the VU's Web site, submit their learning needs, specifying the general or exact topic they are interested in. They will give their preferences for language, length of course, time and medium of delivery. They can also supply the VU with personal profiles detailing their backgrounds and proficiency levels and outlining their professional goals, if they so desire. In return, the VU will guide the customer to the suitable electronic catalogs to assist in course comparison and selections. If convinced, the customer will proceed to electronic payment for the educational products, and participate in the course online with continuous fulfillment over the Internet, through some specially adapted approaches and unique resources assembled for the homogeneous group taking the same course. In practice, the VU is expected to draw on the standard b-c EC methods to deliver information over the networks. These could include ensuring the security of the information; carrying out transaction processing and electronic payments; and routing the traffic to the appropriate secure Internet servers. Security and authentication mechanisms are of high priority because they are needed to guarantee the integrity of the learning materials and to ensure that only authorized customers have access to the materials and examinations, and receive credit on completing the requirements. Besides, development of electronic payment systems based on a wide variety of industry-accepted standards, and a flexible payment scheme (installment possible) is also important from the supplier's perspective. Certainly, it is in the interest of both the customer and the VU to establish standards and quality requirements that apply to the electronic educational product and

define procedures for the certification and assessment of the learners' progress and achievement.

10.2 VU's e-Business Model for the B-C EC Scenario

The VU's fundamental business problems could be stated as follows. We must have something to sell, make it known to potential customers, induce the buying action, accept payment, deliver the goods or service, and provide appropriate fulfillment service after the sale. Also, we want to create a customer relationship that will bring repeat business. This whole sequence of ideas or actions is generally known as the commerce value chain [21]. At each step in the chain, something of value must be added along the way in creating and delivering the product. When applied to the VU's b-c context, the value chain could include: selecting the educational products (degree programs + courses) that will be offered; making or purchasing the course elements for course/program packaging; arranging an attractive electronic display; advertising to attract potential students; assisting customers with their selections; taking electronic payments for the product; and delivering the course to the customer. Each of these links in the chain is important to the business, and if any of them breaks down, the whole business is affected.

10.3 VU's e-Process Model for the B-C EC Scenario

Indeed, the VU's e-Processes can be conceptualized as a basic combination of four core customer-oriented activities: Attract, Interact, Act, and React. These are considered by Treese and Stewart [29], as the generic value chain in developing a business system for Internet commerce:

- *Attract customers.* This is the marketing effort to get and keep customer interest. It refers to whatever steps the VU takes to draw customers into the primary Web site, whether by paid advertising on other sites, email, television, print, or other forms of promotion. The idea is to make an impression on customers and pull them into the detailed catalog or other information about products and services for sale.

- *Interact with customers.* This is the sales effort to turn interest into orders. It is generally content-oriented and includes the catalog, publications, or other information available to customers over the Internet. Technically, the content may be static or dynamic, depending on the respective editorial requirements to change it infrequently or frequently.

- *Act on customer instructions.* This is the function of order management. Once a buyer has searched through a catalog and initiates to make a purchase, there must be a way to capture the order, process payment, and handle fulfillment. There must be some convenience of purchase installed, say, providing the shopping cart for customers to modify purchase content, to discard items, to add new ones, to change quantities, and to compute the total costs.

- *React to customer requests.* This is the function of customer service in the form of technical support. After a sale is complete, the customer may have questions or difficulties that require service. Usually, companies provide the facilities of help desk or service center to handle customer inquiries. Though many questions require a person to answer immediately, oftentimes many can be handled with an appropriate information system, like a Web-enabled learning (course support) environment in the VU context.

10.4 VU's e-Application Model for the B-C EC Scenario

The b-c e-Applications that support the b-c purchasing process, would have to address and provide functionality consistent with the following phases: product identification, catalog search, product comparison, and purchase.

- *Product Identification.* This phase of the b-c e-Application often involves a combination of online advertising and one-to-one marketing in virtual marketplaces. On the Internet, advertising opportunities include banner advertisements on Web sites as well as inclusion in important content aggregation sites such as certain search engines, and aggregator portals. Once advertising has driven traffic to the VU's site, one-to-one marketing and content personalization become key factors. Frequently, marketing promotions aim to encourage

buyers to not only make individual purchases (join certain online programs or individual courses), but also develop long-term business relationships with the university.

- *Catalog Search.* This phase focuses on the product catalog itself, which is the primary repository for product and service offerings. The b-c e-Application needs to construct an electronic customizable catalog aimed to provide goods and services to potential customers. Most b-c catalogs include products and services with pre-negotiated prices, and they should be easy to navigate, enabling the customers to browse specific areas they choose. Mostly, in order to bring back customers for repeat business, the e-Application will try to tailor content by tracking their browsing and generating content to fit their suspected tastes.

- *Product Comparison.* This phase provides features that enable customers to navigate catalogs, compare similar offerings, and make product selections. The most complex catalogs dynamically generate catalog content from a collection of multiple supplier catalogs in order to facilitate and automate the comparison process. A successful electronic catalog should be flexible, permitting the VU to change its catalog sources as the market shifts and to cater to a customer base with different interests.

- *Purchase.* When a potential customer is finally ready to make a purchase, the e-Application must provide authentication and encryption services to ensure that the transaction is confidential and accurate. The VU must be prepared to accept some form of online credit, such as credit card or digital cash. Once a transaction is processed, details about individual product preference and browsing habits can be stored under a personal profile, ensuring that the customer next shopping experience has a one-to-one feel. In the VU context, as many students require at least some student services in the course-taking phase, the b-c e-Applications should hand the customer data off to a student care management application before ending the purchasing transaction.

10.5 VU's Technology Components for the B-C EC Scenario

The technology components employed to translate the VU's b-c EC strategies into effective e-Applications, are based on the underlying business processes (e-Process model), which often require defining specific entities such as customer, catalog, program, course, materials (based on the e-Application models). Recently, component-based EC applications development has become the current trend, where EC systems consist of a lightweight kernel to which new features can be added in the form of business components [3]. In the software industry, components are referred to as a specific piece of functionality that can be accessed by other software through a contractually specified interface. They are self-contained, clearly identifiable artifacts that describe and perform specific functions [24]. And the business component concept is developed to meet an architectural vision where components can be put together to form complete business systems – an environment where components combine dynamically to implement the required business processes and where new components are added to deliver new business functions.

- *e-Application Rules.* When a situation covered by the business rule occur, an obligation is incurred to satisfy the rule. Typically, this is achieved by the *process* business components and the *entity* business components [9]. The former represent the business processes identified in the problem domain, and they are subject to change based on the business rules because business processes are the most volatile aspects of a business system. The latter represent the relatively stable business entities identified in the same problem domain, and they are responsible for achieving the designated objectives by performing the steps of business processes. Principally, the basic capabilities of business components should include the plug-and-play features at various granularities, interoperability across networks, portability on different hardware and software platforms, coexistence with legacy applications, mobility in various networking environments such as Internet, Intranet, and Extranet, and ability of self-managing data resources.

- *e-Application Data.* The enterprise data for the b-c EC scenario includes the data that are shared across the business processes of the VU. It mainly includes the customer information used in different applications such as order processing, billing, accounts receivable/payable, and marketing. Typically, this data has to be architected (i.e., modeled, designed, allocated, interconnected) and

managed carefully [30]. Essentially, the enterprise data is considered as an enterprise asset, and not a single application captive property. Architecture of the b-c EC data, involves a combination of issues. These include: classical relational database design issues (data modeling, logical database design, data normalization); distributed database design issues (data partitioning, data allocation); and middleware issues such as selecting appropriate client/server paradigms and protocols (remote data access or remote procedure call) and database gateways. The database life cycle ties these issues and approaches into a consistent framework, which has become an important technology component of the VU's reusable asset repository.

- *e-Application Distribution.* The distribution model of the b-c EC e-Applications lies on the separation of business rules (functional logic) from the presentation (user interface) and data management functions of applications. This separation of concerns leads to a three-layered application architecture: presentation services, data services, and processing services. Such separation promotes the notions of interoperability, reuse, and manageability of applications. Different modules can be designed and deployed on different machines for different presentation styles, user groups, and data management software, according to a specific distributed architecture. For example, two-tiered architectures commonly refer to the traditional client/server (C/S) applications that use remote SQL to access database servers (tier 2 machines) from programs and user interfaces residing at desktops (tier 1 machines); and three-tiered architecture introduces a middle machine dedicated to application logic [8].

11. The VU's Software Architectural Approach to Reuse

The VU's reuse policy is conceived around an architectural approach to software design, offering such promising benefits as the reduction in costs of development, and the increase in reliability [1]. Architecture-driven design is aimed to fulfill not merely the functional requirements (size and performance), but also the non-functional aspects such as ease of maintenance, and accommodation of projected changes. It carries out design within the context of the architecture. Currently, many systems were designed and built individually, and previous "similar but different" systems were examined to see if anything could be taken out and used again under the same context. We believe this approach is not systematic because the strategy is one of opportunistic and accidental reuse. For reuse to be systematic, it needs to be planned for. If we produce an architecture for a particular context, and design some code within the constraints of the architecture, then it is believed that wherever we use the architecture, we can reuse the code. Actually, architecture-based reuse represents more than planning for the systematic reuse of code, but also that of the context referring to the reuse of analysis and design, which defines how function is located and how functional loci communicate. Yet, abstracting common context for a range of systems is difficult, and it can hardly be found without example systems to work from. We believe that the object technology could help since it has a heavy emphasis on the process of finding good abstractions. Besides, an architecture relies on an architectural vision, and it is about structure rather than function. If we choose an architecture that meets non-functional requirements, it shall constraints the design work, which is required to meet the functional ones. This is believed to be a more mature process than just carrying out object-oriented analysis, design and implementation. Overall, we believe that architecture is the key to reuse. A common architecture can be used as the basis for a number of similar-but-different systems. This means we are also reusing analysis and design and so achieve greater levels of reuse than if we were reusing code alone. This also represents our emergent philosophy of how to develop and evolve our business and technology architectures in the VU context.

12. The VU's WIS Applications Engineering

Earlier we mentioned that the WISs are interpreted as an important iterative and incremental means to realize the information system services offered dynamically by the VU according to its ongoing business requirements. These include: increased demands for flexibility, pressures to respond quickly to market conditions, intense local and global competition, and continual business process re-engineering and improvement for enterprise efficiency. In this section, we discuss the construction of individual WIS applications based on the interrelationships among business processes, applications, and IT platforms. Business processes represent the day-to-day business related activities of

the VU. They support enterprise goals and thus establish requirements and the business drivers for lower level technology utilization. Applications provide automated support to the business processes, and thus provide business value to an enterprise. They are said to be business aware in a sense that they entail business logic and data that is specific to the business requirements. They typically consist of a user database, a set of programs to access and manipulate the database, and user interfaces to execute the programs. The IT platforms are used to build, deploy and operate the applications. They enable the applications and are business unaware in a sense that the same platforms could be used for different business systems. An important part of the IT platform is middleware [2], which is used to interconnect and support applications and users across computer networks. Middleware services typically include directories and facilities to call remotely located procedures and software to access and manipulate remotely located databases. Specifically, we believe that business processes provide the requirements that drive the applications and the IT platforms. And applications enable the business processes and the IT platforms enable the applications. Essentially, IT platforms do not add any direct business value, and if not handled properly, could disable applications and business processes. In the VU context, a key role of the IT platforms is to enable various distributed applications to support the distributed business processes.

12.1 Developing WIS Application Architectures

Development of a WIS application is composed of a series of mini applications, which are closely tied together to solve a business problem. Different parts of this application may support different classes of users, utilize different programming languages, run at different sites, use different computing platforms, employ different database technologies, and be interconnected through a diverse array of network communication devices and software. We need a systematic methodology to develop WIS applications that maximize the benefits to the VU, and minimize the risks [13]. And it is convinced that such methodology consists of successive iterations, refinements, and expansions between analysis, architectures, implementations, deployment and support activities. Here we focus on describing the architectural concerns in constructing the WIS application, which depict how the mini-systems tie together to satisfy the necessary requirements. Simply put, the architecture of a WIS application should describe the components, the functions of the components, and the interactions between the components of the application. More precisely, a WIS application requires the development of an application architecture involving respectively its data, software and IT-platform components, which are commonly referred to as its data architecture, application software architecture, and platform architecture. And they are the functional equivalents of our respective constituents (e-Application Data, e-Application Rules, e-Application Distribution) in the VU's technology architecture previously mentioned.

12.1.1 Data Architecture. This architecture is concerned with the crucial issues of data definition, data placement, data access, and data administration. The definition of data identifies such concerns as what constitutes the required data, who owns them, who generates them, and who will use them. The placement of data involves such concerns as what data are located where in the enterprise network (Intranet, Extranet), how many copies of data exist, where they exist, and which copy is primary. The mechanism for data access points to who needs to access the data, on what platforms they should reside, whether or not the users employ the Web or other technologies to access the database. The administration of data manages who is responsible for the integrity, security and quality of data.

12.1.2 Application Software Architecture. This architecture catalogs the application components, manages the inter-relationships between these components, administers their allocations and their coordination paradigms. It raises the portability and interoperability issues in distributed environments because numerous mini-applications are tied together within this architectural context to fulfill different business and technical requirements. According to Umar [30], application architecture development involves the following steps: a) define a logical model (often an object model), which describe static and dynamic behavior of the application; b) extend the model to a distributed object model by dividing the application logic between client and server objects and evaluate issues such as two- versus three-tiered configurations, as well as thin-client or fat-client approach; c) evaluate/choose the IT platforms, especially the middleware; and d) evaluate the allocation strategies of application components to

various computers (hardware tiers) and analyze performance trade-offs.

12.1.3 Platform Architecture. This architecture provides the set of technologies (middleware, networks, operating systems, database/transaction managers, computing hardware) that glue together the application pieces across an enterprise. A typical WIS application may use a combination of middleware such as Web technologies (Web browsers, Web servers, HTML + XML, and HTTP), distributed object technologies (CORBA, DCOM) and SQL middleware (ODBC + JDBC drivers, database gateways) that operate over TCP/IP networks across PC Windows, Windows NT, UNIX, and LINUX platforms.

12.2 Deriving VU's Enterprise Information Architecture

In engineering the WIS applications, it is important to derive the VU's enterprise information architecture (EIA), which specifies the IT-based solution approach that will satisfy the information requirements of the enterprise [5, 16]. The EIA is aimed to specify those pieces of information (enterprise data, applications, and platforms) which should maximize benefits and minimize costs and risks. Particularly, it should minimize the impact of platform and vendor changes on applications. Technically, this means promoting the use of tools that should hide the underlying middleware details (application programming interfaces, APIs) from programs. Principally, the VU's EIA is composed of its business architecture and technology architecture. The latter is often expressed in terms of the various application architectures including respectively the specific data architectures, application software architectures, and platform architectures of the various WIS applications. Operationally, once an overall EIA is in place, the selection and modification of the platforms should not affect the implemented application whose investment is thus protected despite platform and vendor changes. Simply put, the VU's EIA provides a framework for reducing information system complexity and enables the enterprise to effectively and efficiently deploy new technology for enterprise information sharing, with the promising benefits of reducing data and software redundancy to move to the new technology.

12.3 Previewing WIS Application Software Architectures

Development of software architecture for WIS applications is quite a complex endeavor. We must take into account the data architectures, the application requirements (functional, operational, organizational), and the platform architectures (ever-evolving infrastructures and standards). So, an approach is needed to decompose the software architecture activity into smaller and manageable steps. Here, we describe in an executive summary manner the important guidelines we have been following as follows [30].

12.3.1 Refine and Extend the Object Model. This step is meant to represent as accurately as possible the abstract components (objects) of the application and the abstract messages between these components. Granularity of components is a major concern: small objects that represent program routines versus large business component that represent business entities. Typically, we use the object-oriented analysis and design techniques to build the abstract model that captures the functional requirements of the application in an implementation-independent fashion.

12.3.2 Separate Concerns into Suitable Software Layers. This step groups the components into three layers: presentation (P), application logic (L), and data management (D). This is based on the separation of business rules from the user interface and data requirements of the application. This separation of concerns promotes the notions of interoperability, reuse and manageability of applications in distributed environments because different modules can be designed and deployed on different machine for different presentation styles, user groups and data management software.

12.3.3 Decompose Application into Tiers. This step casts the logical architecture into a client/server configuration which identifies the physical tiers (physical levels of distribution) of the application. An application can be physically configured as: single-tiered, two-tiered, three-tiered. The first assigns all application layers (P, L, D) in one machine. The second splits the application layers between front-end and back-end computers. The

former typically includes the P + L layers (thick client), whereas the latter the D layer (thin server). The third splits the application layers across three types of machines: a front-end with the P layer (thin client), a middle machine with the L layer, and a back-end machine with the D layer. Determination of an appropriate level of distribution impacts the choice of infrastructure, especially middleware, performance results and implementation considerations of the application.

12.3.4 Evaluate/Choose an Infrastructure. An infrastructure is needed to support interoperability, facilitate different interaction paradigms, enable system management (security, failure handling, performance management), and ensure needed network services for connectivity between remotely located client and server processes. This step requires examination of a wide range of issues such as APIs, exchange protocols, data management software, directory services, and network services. It is a challenging step, especially when numerous distributed WIS applications span multi-vendor environments.

12.3.5 Performance and Allocation Analysis. Allocation of application components to different machines impacts the performance of distributed applications. A well-architected application can easily self-destruct if its components are assigned to slow or congested computers that are interconnected through slow communication links. It is important to optimize the performance of applications by placing application components (databases and programs) at the most appropriate sites. This step is to determine the best sites for allocation of data, processes and user interface handling.

12.4 Architectural Standards for WIS Applications Components

Given the architectural concerns described above, software developers are faced with significant challenges as they attempt to mold their current solutions into real implementations. In this section, we shall discuss some of the emerging technology components that are the underlying implementation infrastructure for the WIS applications. We call them standards because they represent well-defined bodies of work intended for widespread use according to its definition, whether promoted by a standardization body, or a commercial vendor or consortium.

12.4.1 CORBA – Common Object Request Broker Architecture. The CORBA standard is conceived around the idea of distributed objects. Essentially, we are talking about decomposing enterprise-wide applications into objects that can reside on different machines on a network. And an object on one machine can send message to objects on other machines, thus viewing the entire network as a collection of objects. Support for these distributed-object-based applications requires special-purpose middleware that will allow remotely located objects to communicate with one another. A common mechanism used by such middleware is an object-request broker (ORB) that receives an object invocation and delivers the message to another appropriate object. And CORBA [25] is the basic Object Management Group (OMG) standard that allows software applications to communicate in this fashion. Specifically, it allows different components of an application to execute in different locations, on different hardware/software platforms, and in different programming languages. It is relatively mature and supported by a large number of vendor products.

12.4.2 UML – Unified Modeling Language. UML [4] is a language for specifying, visualizing, constructing, and documenting the artifacts (architecture) of software systems, as well as business modeling. It represents a collection of best engineering practices that have proven successful in the object-oriented modeling of large and complex systems. The development of UML had many objectives in mind including to provide sufficient semantics to address certain future modeling issues related to distributed computing, frameworks and component technology, and to facilitate model interchange between a variety of tools. The basic UML semantics can be used to define classes, interfaces, attributes, operations, relationships, state machines, use cases and many other details of object-oriented models. It also provides extensibility mechanisms that effectively make it into a family of related languages, each customized for a specific purpose. Historically, UML is developed by OMG, near the end of 1997, to serve as the standard language of blueprints for software.

12.4.3 XML – eXtensible Markup Language. XML [23] is a new tag-based language for describing data. It is basically a subset of the Standard Generalized Markup Language (SGML), a complex standard for describing document structure and content. XML, being a meta-language, lets us define our own customized markup languages for different document classes. It is defined by the World Wide Web Consortium (W3C) to ensure that related data can be transmitted and exchanged in a structured manner over the Web. Unlike the Hypertext Markup Language (HTML), which is being used to convert text stream into the presentation of a page, XML's goal is to convert text stream into a data object with a complex inner data structure and leave the presentation separate. This separation of structured data from presentation allows seamless integration of data from diverse sources. Purchase orders, catalog data, and other similar information from disparate sources can easily be converted through the Web browser or could be fed to other distributed application for further aggregation and processing.

12.4.4 XMI – XML Metadata Interchange. XMI was adopted as an OMG standard in March 1999 [27]. It gives a standard method to interchange UML models in the form of XML documents, that can be stored in files and streamed across the Internet. XMI is based on an OMG standard called MOF (Meta-Object Facility) which is used to define meta-models. This means we can use XMI to interchange models in any modeling language defined using MOF, of which UML is just one example. UML and XMI together provide the capability to define semantically rich object models in a tool-independent format, and the ability to interchange such models between one tool and another without loss of information.

12.4.5 EJB – Enterprise Java Beans. This is a standard programming model for enterpise componentry, and is member of a set of technologies and tools defined by the JavaSoft [26], for developing multi-tier distributed applications. The infrastructure is called the Java Platform for the Enterprise (JPE), which besides EJB, also includes Remote Method Invocation (RMI), Java Naming and Directory Interface (JNDI), Java Database Connectivity (JDBC), Java Transaction Service (JTS) and Java Messaging. Basically, an EJB is a software component that defines business application logic, and is run on an Enterprise Java Server (EJS) which implements containers. Containers are conceived as software constructs, where EJBs are deployed, and which implement 'qualities of service' for the EJBs to utilize as they executes, such as transactional behavior, security characteristics, and persistence features. The EJB specification defines two kinds of beans: session beans and entity beans. The former are intended to implement a piece of business logic on behalf of a single client. The latter are representations of a persistent state, normally in a database, or implemented by an existing enterprise application, to be shared between multiple client sessions. Hence, the EJB model is enabled to implement a complete separation of business logic from technical processing.

13. Remarks for Continuing Challenges

The overall picture confronting today's enterprises is this. At the core is the installed base of existing IT systems, which includes the legacy data and business logic. Around the edge are increasingly pro-active customers, to which the enterprise must offer an increasing quality of service through existing and new channels. In between, the enterprise is re-engineering its business processes, with a focus on knowing its customers better, and offering continuous improvement of its products and services. From an IT perspective, the legacy systems become surrounded by a matrix of go-between componentry providing services to support the changing business, with increased flexibility and reduced development times as compared with the legacy systems. This is the backdrop behind which our VU is conceived in terms of its WIS applications development. And we call this, the VU's e-Transformation which is tackled based on the virtual organizational model focusing on a categorical split between task requirements and their satisfaction. This paper has focused on identifying the abstract requirements needed to realize the WIS applications, while at the same time, investigating and specifying the current technological means (software architectural approach) for satisfying the requirements. Among the numerous WIS application design problems to be solved include the following. For ease of development and reuse and interoperability, we need to develop some architectural descriptions (perhaps based on UML, CORBA, XML) to define some reference architectures to address the specific Web-development details of the WIS. For example, a collaboration architecture could provide, to a dispersed set of participants, an effective means of

working together and communicating with one another on topics of joint interest. A thin-client transactional architecture could address the need to do enterprise-wide administrative business. A service center architecture could provide Web-based access for customers to do business, say, accessing the digital learning environment in the process of serving the student client. Such reference architectures may be thought of as large-scale design patterns [6, 17], which should capture the design of enterprise-wise WIS applications.

References

[1] Anderson, B., and Dyson, P., "Reuse Requires Architecture," In L. Barroca, J. Hall, and P. Hall (Eds.), Software Architectures: Advances and Applications, Springer-Verlag, Berlin Heidelberg, 2000, pp. 87-99.

[2] Berstein, P., "Middleware: A Model for Distributed System Services," Comm. ACM, Vol. 39, No. 2, Feb. 1996, pp. 86-98.

[3] Bichler, M., Segev, A., and Zhao, J., "Component-based E-Commerce: Assessment of Current Practices and Future Directions," ACM SIGMOD Record, Vol. 27, No. 4, Dec. 1998, pp. 7-14.

[4] Booch, G., "UML in Action," Comm. ACM, Vol. 42, No. 10, Oct. 1999, pp. 26-28.

[5] Cook, M.A.. Building Enterprise Information Architectures: Reengineering Information Systems, Prentice Hall PTR, 1996.

[6] Cook, S., "Architectural Standards, Processes and Patterns for Enterprise Systems," In L. Barroca, J. Hall, and P. Hall (Eds.), Software Architectures: Advances and Applications, Springer-Verlag, Berlin Heidelberg, 2000, pp. 179-190.

[7] Dennis, A.R., "Lessons from Three Years of Web Development," Comm. ACM, Vol. 41, No. 7, Jul. 1998, pp.112-113.

[8] Dickman, A., "Two-Tier versus Three-tier Apps," Information Week, Nov. 13, 1995, pp. 74-80.

[9] Eeles, P., "Business Component Development," In L. Barroca, J. Hall, and P. Hall (Eds.), Software Architectures: Advances and Applications, Springer-Verlag, Berlin Heidelberg, 2000, pp.27-59.

[10] Hamalainen, M., Whinston, A.B., and Vishik, S., "Electronic Markets for Learning: Education Brokerages on the Internet," Comm. ACM, Vol. 39 No. 6, Jun 1996, pp. 51-58.

[11] Hardwick, M., and Bolton, R., "The Industrial Virtual Enterprise," Comm. ACM, Vol. 40, No. 9, Sept. 1997, pp. 59-60.

[12] Harrington, J. Organizational Structure and Information Technology, Prentice-Hall International, Hertfordshire, U.K., 1991.

[13] Henderson, J., and Venkatraman, "Strategic Alignment: Leveraging Information Technology for Transforming Organizations," IBM Systems Journal, Vol. 32, No. 1, 1993, pp. 4-16.

[14] Hoque, F. e-Enterprise: Business Models, Architecture, and Components, SIGS Cambridge, 2000.

[15] Isakowitz, T., Bieber, M., and Vitali, F., "Web Information Systems," Comm. ACM, Vol. 41, No. 7, Jul. 1998, pp.78-80.

[16] Melling, W., "Authoring an Enterprise Information Architecture: A Pragmatist's Approach to a Process," Gartner Group Conference, Feb. 22-24, 1995.

[17] Mowbray, T., and Malveau, R. CORBA Design Patterns, John Wiley & Sons, New York, 1997.

[18] Mowshowitz, A., "Virtual Organization," Comm. ACM, Vol. 40 No. 9, Sep. 1997, pp. 30-37.

[19] Mowshowitz, A., "Virtual Organization: A Vision of Management in the Information Age," Inf. Soc. 10, 4 (1994), pp. 267-288.

[20] Mowshowitz, A., "On the Theory of Virtual Organization," Syst. Res. 14, 4(1997).

[21] Porter, M.E. Competitive Advantage: Creating and Sustaining Superior Performance, The Free Press, New York, NY, 1985.

[22] Riggins, F.J., and Rhee, H.S., "Toward a Unified View of Electronic Commerce," Comm. ACM, Vol. 41 No. 10, Oct. 1998, pp. 88-95.

[23] Roy, J., and Ramanujan, A., "XML: Data's Universal Language," IEEE IT Professional, Vol. 2, No. 3, May/Jun. 2000.

[24] Sametinger, J. Software Engineering with Reusable Components, Springer-Verlag, Berlin Heidelberg, 1997.

[25] Seetharaman, K., "The CORBA Connection," Comm. ACM, Vol. 41, No. 10, Oct. 1998, pp. 34-36.

[26] See http://www.javasoft.com.

[27] See http://www.software.ibm.com/ad/features/xmi.html

[28] Takahashi, K., "Metalevel Links: More Power to Your Links," Comm. ACM, Vol. 41, No. 7, Jul. 1998, pp.103-105..

[29] Treese, W., and Stewart, L. Designing Systems for Internet Commerce, Addison Wesley, Reading, Mass., 1998.

[30] Umar, A. Application (Re)Engineering: Building Web-Based Applications and Dealing with Legacies, Prentice Hall PTR, 1997

[31] Weston, R., and Nash, K., “Intranet Fever,” ComputerWorld, Vol. 30 No. 27, 1996, pp. 1-15.