Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2007 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2007

Web-Based Information Systems: Developing a Design Theory

Keng Siau
University of Nebraska-Lincoln

XiaoFeng Chen

Xin Tan
Fairleigh Dickinson University

Recommended Citation

Siau, Keng; Chen, XiaoFeng; and Tan, Xin, "Web-Based Information Systems: Developing a Design Theory" (2007). *AMCIS 2007 Proceedings*. 283.
http://aisel.aisnet.org/amcis2007/283

Web-Based Information Systems: Developing a Design Theory

Keng Siau
University of Nebraska - Lincoln
ksiau@unl.edu

Xiaofeng Chen
University of Nebraska - Lincoln
xchen@unl.edu

Xin Tan
Fairleigh Dickinson University
xtan@fdu.edu

Abstract

Design theory provides a description of the method and/or structure for the construction of an artifact. In the IS field, design theory represents one of the five major types of theory (Gregor 2006).Unlike theories for analysis, explanation, or prediction, IS design theories take a prescriptive approach by providing principles, functions, methods, and theoretical knowledge that are used in the development of IS. Design principles for Web-based systems are scattered and limited in the IS literature. The uniqueness of Web-based systems development warrants a new design theory for such development efforts. Based on an action research study, a set of 10 design principles were proposed.

Key words: Design Theory, Information Systems Development, Web-based systems

Introduction

With rapid advances in information and communication technologies, an increasingly important research topic in the information systems (IS) field is how to effectively develop new systems. Many development methodologies have been devised to improve the management and control of development projects, and to standardize the development process and product. Walls et al. (1992) used the term "IS design theories" to refer to an integrated prescription for IS development efforts. Using this term, examples of design theories include IS development methodologies, for instance the Software Development Life Cycle model and Unified Process, and prescriptive guidelines for developing specific applications, for example knowledge management systems in Markus et al. (2002) and executive information systems as in Walls et al. (1992). IS design theories make many contributions to the IS field. First, an IS design theory articulates the boundaries within which particular design assumptions apply (Walls et al. 1992). Second, IS design theories help make the development process more tractable, and ultimately improve development outcomes (Markus et al. 2002).

Following the call for articulating and developing a class of design theories (Walls et al. 1992), and in the spirit of Markus et al. (2002), this paper proposes a new IS design theory for Web-based systems. A design theory is needed for web-based system because existing research in web-based systems development, either from development methodology perspective or from human-computer interaction (HCI) perspective, is isolated, unable to provide a comprehensive yet applicable set of guidelines. In addition, some features in the process of developing web-based systems are unique, which make existing design theories for other types of IS inappropriate or inadequate.

The rest of the paper is organized as follows. In the theoretical background section, we introduce concepts associated with IS design theory and review related literature to explain why a IS design theory is needed for web-based systems. Next in the research method section, we present the action research study. Then the design theory in terms of a set of development principles is presented and discussed in detail. Finally, we discuss the future research and the study's contributions for both researchers and practitioners.

Theoretical Background

In this section, we first introduce the concept of IS design theory. Then, we review prior literature on Web-based systems to reveal the need for a new design theory.

Introduction to IS Design Theory

A recent review (Gregor 2006) indicates that there are diverging views on the status of design theory and its relationship to other types of theory in IS field. Gregor (2006) found that relevant studies often appear in different labels, such as software engineering research, constructive type of research, system development approach, and design science. Walls et al. (1992) clarified the nature of IS design theory as "a prescriptive theory based on theoretical underpinnings which says how a design process can be carried out in a way which is both effective and feasible" (p.37). This perspective is shared by Markus et al. (2002), who stated, "IS design theories are intended to give guidance to developers."

Markus et al. (2002) summarized the design theory as consisting of three interrelated elements: (1) a set of user requirements derived from kernel theory, (2) principles governing the development process, and (3) principles governing the design of a system (i.e., specifying and implementing its features). Whereas many of the IS courses we teach focus on design and developing information systems, research in design-related issues is scattered and under represented in IS literature (Bajaj et al. 2005).

Prior Work in Web-based Systems

Although web-based systems have been in existence and widely adopted since mid-1990s, research in Web-based system, especially from a design science perspective, is scattered and limited in the literature. Prior work in web-based system was mainly concentrated in development methodology or web site usability from HCI perspective. However, the use of methodology in small application development team is not popular. Whitley (1998) pointed out that integrated system development methods do not necessarily lead to good understanding of the situation developers are investigating and methodologies are often ignored in systems development.

Studies on web usability from HCI perspective are extensive in the IS literature. Due to the length limitation for the proceeding, we will not discuss the findings in detail in this paper. Nielsen (2000) in his classic book about web usability emphasized the importance of simplicity of web site design: fast response times are the most important criterion for Web pages.

From the development methodology perspective, there are some studies on the difference between traditional application development and web-based application development. For example, Chau et al. (2003) studied a multi-agent collaborative web mining system. Janicki et al. (2001) researched on development and evaluation of a framework for creating web-based learning modules.

While these two streams of research have provided some guidance to web-based systems development, their separation from each other limits their utility for the development community. On one hand, the proposed development methodologies mainly focus on the primary activities involved in Web-based systems development, such as requirements determination and analysis. On the other hand, the findings from web usability studies usually emphasize on limited aspects of web page design.

The Need for a New Design Theory

Although numerous web-based systems have been developed since late 1990s and early 2000s, design science research regarding this IT artifact is scarce. The main reason for a new design theory for web-based system development is that the process of web-based systems development is unique. Baskerville & Pries-Heje (2004) identified 10 unique properties in web-based system development through three longitudinal case studies. Those characteristics are: 1) Time Pressure; 2) Vague Requirements; 3) Prototyping; 4) Release Orientation; 5) Parallel Development; 6) Fixed Architecture; 7)Coding Your Way Out; 8) Quality Is Negotiable; 9)Dependency on Good People; 10) Need for New Kinds of Structure.

We identified three other features in web-based systems development. They may not be unique in web-based systems development, but exhibit higher intensity in web-based systems development than in traditional systems development. The first is great heterogeneity in users, developers, and requirements; the second is the need for web-based systems to be closely integrated with other enterprise systems and partners' information systems; and the third is constant requirements changes requiring frequent maintenance and upgrade by the developers.

Because of the uniqueness of web-based systems development, a new design theory is warranted to guide the process of web-based systems development.

Research Method

Research Approach: Action Research

Baskerville & Wood-Harper (2004) stated that "Action research is an interventionist approach to the acquisition of scientific knowledge that has sound foundations in the post-positivist tradition." Davison (1998) summarized the role of action researcher in an action research from various studies and presented that "action researchers either participate directly in or intervene in a situation or phenomenon in order to apply a theory and evaluate the value and usefulness of that theory (Checkland 1981, 1991; Argyris and Schon 1989; Dick 1993; Vreede 1995; and Davidson 1998)". Galliers (1991) pointed out that action research can be used for theory testing and theory building and Wall et al. (1992) stated that action research is appropriate for design theory building.

We believe action research is an appropriate research approach for this study. Action research has various forms and characteristics (Baskerville 1999). Our study was guided by Baskerville (1999)'s participatory action research. Other than regular characteristics of action research, Baskerville (1999) pointed out that participatory action research is distinguished by the additional characteristic involvement of the practitioners as both subjects and co-researchers. An important change in participatory action research from traditional action research is "the realignment of the roles of researcher and subject into more collaborative and synergistic forms."

One author was actively involved in the development project, which served as the research site for this study. In the following, we briefly describe the development project.

Research Site

At a mid-west public university, a web-based system was created in 1998 using a platform called Uniface from Compuware. The objective of developing this web-based administrative system (WAS) is to provide convenience to students, reduce the costs in the university administrative offices, and increase the efficiency of the university in admission, registration, and student accounts offices and ultimately increase the student's satisfaction toward the administrative processes in their campus lives.

A web-based system was new to the developers in the IT department at the university. The developers had worked in a mainframe environment and on client-server application exclusively. Their top priority was and still is production support. Most of their time was allocated to keep the current production system work and keep it enhanced and modified to meet the always-changing business requirements. It is a big challenge for those developers to learn a new platform on which new systems are developed. It is estimated at one unit of the university's central computing department that the developers at that unit can only have about 20% of their time for product development.

The constant learning for developers was heightened when the system was re-written in Java in the last two years. The reason for the rewriting was that Uniface is a proprietary development tool and server environment and Compuware was planning to stop the support for this technology Compuware was trying to switch its effort to develop tool-sets from Uniface for Java technology. During this rewriting of the old system, many more functions were added into the system.

Unified Process as Development Methodology

Unified Process (UP) was adopted as a development methodology for the development team. UP is the de facto process for software development. We tried to use a flexible methodology that can be tailored to fit the needs and requirements of the organization. UP is an iterative and incremental process. UP is use-case driven, architecture centric, and risk-focused. UP is not a single concrete prescriptive process, but rather an adaptable process framework, intended to be tailored by the development organizations and software project teams that will select the elements of the process that are appropriate for their needs.

UP has four phases: Inception, Elaboration, Construction, and Transition. In the inception phase, business context is defined and success factors are established. In Elaboration phase, problem domain is analyzed, use cases are created, and software architecture is proposed. In Construction phase, the focus goes to development of designed components. In Transition phase, main tasks are training of end users and maintainers and beta testing of the system to validate it against the end users' expectations.

The problems encountered in using UP in the development of the web-based systems include:

- UP provides a set of generic guidelines and does not specify how to do things. It specifies what need to be done, not how to do.
- UP, as a complicated methodology, is difficult to follow without specific design guidelines.

Therefore, prescriptive guidelines from a design theory can supplement UP in guiding the development effort. UP was used as a methodology for the development team. It is also used as our research framework to organize our research findings in terms of design and development principles.

Proposing a design theory for Web-based systems

In this study, we follow guidelines for action research and case study methods in proposing a design theory for Web-based systems. Based on the collected data, the researchers conducted qualitative analyses to reach initial list of design principles for web-based systems. Through discussions, the researchers finally agreed on a set of 10 principles. In the following, we describe these principles framed in the four phases of UP.

Principles for Inception Phase

Principle #1: Creative strategy for domain expert engagement

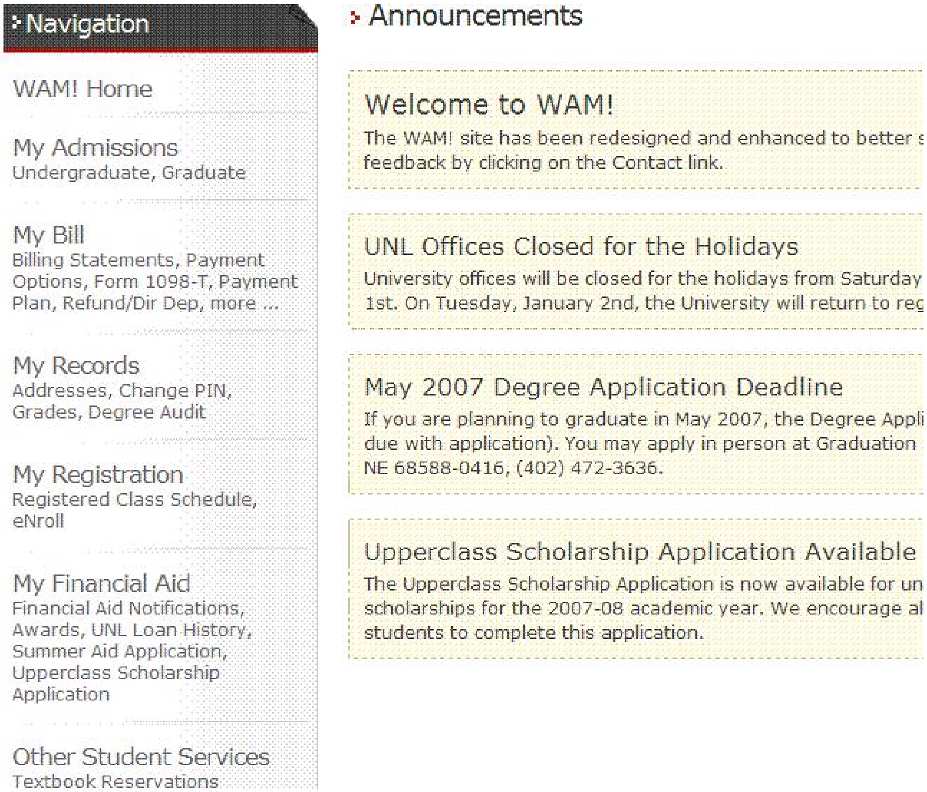
One of the problems in business application development is domain expert engagement in gathering application requirements and application testing. Engaging domain experts in the development lifecycle is a critical factor to ensure the success of a IS/IT project because they can contribute greatly in requirement development and in application testing. Getting domain experts involved in project development may have critical implication for finishing the project on time and on budget.

Therefore, it is important to develop strategies to get domain experts involved in your project development from the beginning to the end. The strategies may include having top administrative involved in the decision stage so that the appropriate domain experts can be assigned to the tasks that are required to be done on time.

Engaging domain experts in the development process can help reduce the problem generated by the characteristic of vague requirements in web-based system development (Baskerville & Pries-Heje 2004). One technique to help the engagement is to use Use-case diagrams and activity diagrams to visualize the requirements and communicate the requirements between the domain experts and system developers.

Principle #2: Organize contents in the way users are familiar with

One of the problems in the Internet era is information overloading, which has raised research interest in Website usability studies. Many usability evaluation methods have also been developed. Norman (1998) in his historic book "Psychology of everyday things" argued four high-level design principles: Propose a good conceptual model, make the things visible, natural mapping, use of feedback. Those principles are good guidelines for the layout of the website. Organizing services around the busines units may be the best choice among different ways of organizing services since it maps to users everyday life in that context (see Figure 1 for the layout).


Navigation
WAM! Home
My Admissions
Undergraduate, Graduate
My Bill
Billing Statements, Payment Options, Form 1098-T, Payment Plan, Refund/Dir Dep, more ...
My Records
Addresses, Change PIN, Grades, Degree Audit
My Registration
Registered Class Schedule, eNroll
My Financial Aid
Financial Aid Notifications, Awards, UNL Loan History, Summer Aid Application, Upperclass Scholarship Application
Other Student Services
Textbook Reservations
Announcements

Welcome to WAM!

The WAM! site has been redesigned and enhanced to better
feedback by clicking on the Contact link.

UNL Offices Closed for the Holidays

University offices will be closed for the holidays from Saturday
1st. On Tuesday, January 2nd, the University will return to re

May 2007 Degree Application Deadline

If you are planning to graduate in May 2007, the Degree Appli
due with application). You may apply in person at Graduation
NE 68588-0416, (402) 472-3636.

Upperclass Scholarship Application Available

The Upperclass Scholarship Application is now available for un
scholarships for the 2007-08 academic year. We encourage al
students to complete this application.

Figure 1: The Navigation of the WAS System

Principle #3: Show real time data

Real time data is one of the good reasons to attract users to use the new system. You need to make every effort to make sure the system can display and update real time data for the end users. Prior to the WAS system, students at the university could not get their grades until they were in their mail two or three weeks after their final exams. With the WAS system, their grades are just clicks away after their instructors have posted their grades. And more with the WAS system, now students can see their bill, class schedules, availability of class, and the status of their applications for admission or financial aids, all in real time. Real time data please students in big ways and help save the university money.

Principles for Elaboration Phase

Principle #4: Build right architecture first and deploy services incrementally

This is a principle for all software development. Although web-based application development is pressured by time, this principle should be followed since it will save time eventually because you do not need reinvent the wheel again. This principle conforms to the unique characteristic of web-based systems development: fixed architecture (Baskerville & Pries-Heje 2004). The three-tier architecture is a general architecture to follow in web-based systems development because of the characteristics of higher degree need of web-based systems integration with other enterprise systems and partners' information systems. We also recommend building a more specific architecture as shown in Figure 3 for a web-based system so that additions of new feature will be easier than without such a structure in place. It is almost certain that your web-based systems will go through multiple re-engineering process and features will be added constantly as we mentioned above that constant requirements change and maintenances are a unique feature in web-based systems.

Taking time at the beginning to plan a right technical architecture will save a lot of time for your organization in the long run; therefore, it might be better to deploy simple and easily developed services at the beginning stage and concentrate on architecture and blueprint of your development and deployment environment. At the beginning, we decided that we would put core services in place before we moved on to develop specific services that are related to different administrative offices. For example, we decided that we needed to centralize the authentication and authorization services so that the developers who are responsible for implementing business logic for different offices will not need to worry these services. We also make data access a common service since an Intranet web-based system is data driven and data transaction is the most common activity in the application services. Figure 2 shows the flow for a user request. The developers need only to concentrate on the development for the 'Business Logic' portion. The presentation portion is regulated by templates, which is described in the principle #6.

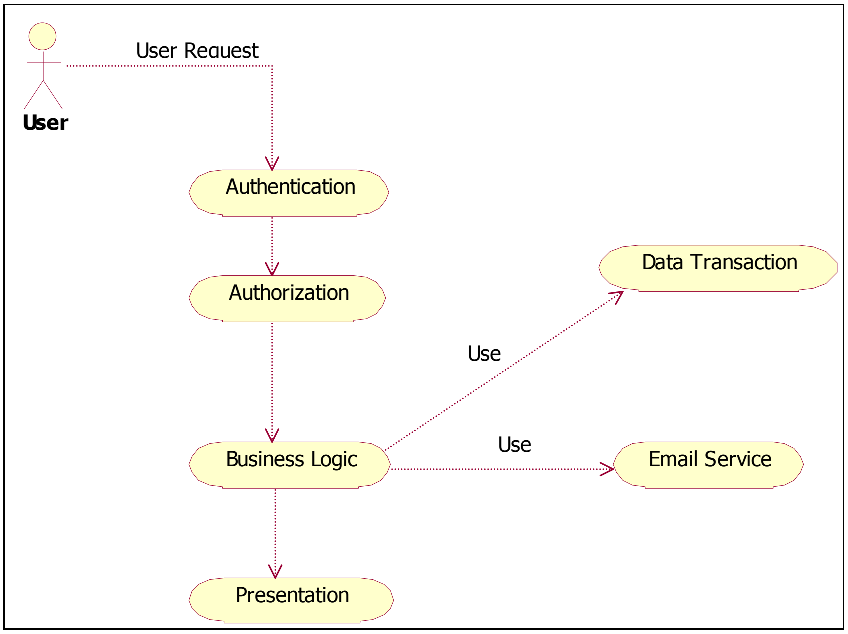


Figure 2: A user request flow in the system

Principle #5: Create your own development procedure and package the key components for all developers to use

The requirements for most of the university's web applications are not mission critical and do not involve complicated transaction processing. However nowadays programming is more complicated than before, requiring programmers to know many more things before they can be productive. The characteristics of "time pressure" and "code your way out" in web-based systems development (Baskerville & Pries-Heje 2004) present another reason for this principle. A customized development procedure and componentized the key packages can assist new developers to start a new project in the right direction and to commit fewer mistakes in the developed product.

The Unified Process used in the development process is hard for developers to follow and practically impossible for a small team to make those artifacts in place. A customized process is essential for the team to have basic structure of the development environment so that every developer knows what to do in each of the development phases.

Componentizing key packages will also benefit the organization greatly when the system needs to change, for example, change data source. Figure 3 shows the packages in the developed system. The first two rows contain the packages that are common for all applications in the system. The third row contains the packages that are needed in each application and it contains business related specifications. When the data source for an application is changed, the only package that needs to be changed for that application is the Data Access Objects package.

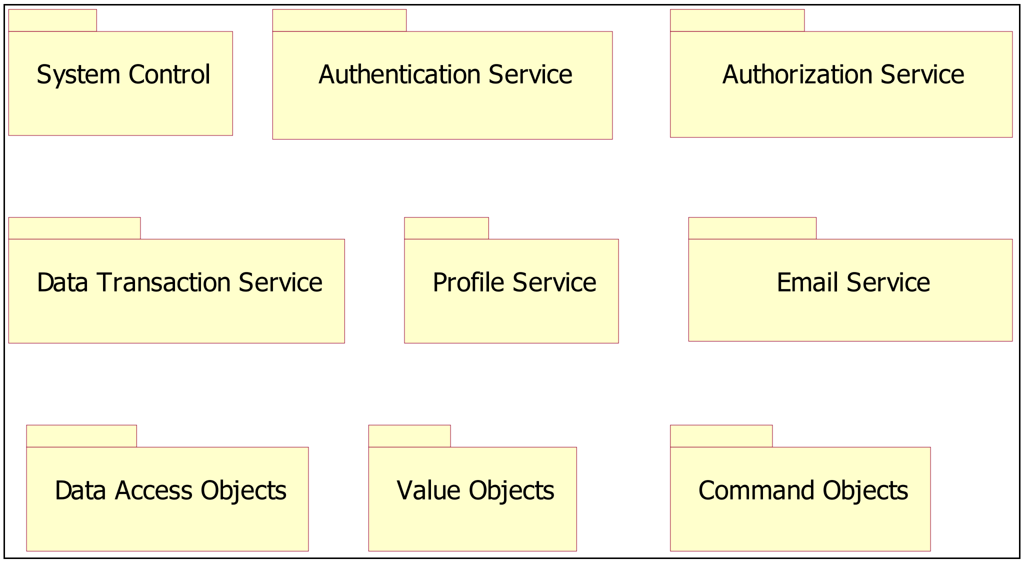


Figure 3: Packages in the Model and Control Layers of the WAS system

Principle #6 Separate business logic, presentation, and control using MVC model

As discussed in principle #5, componentized key packages not only help developers to develop new application faster but also will save time and resource during the system changing process. The key to make componentized packages is the separation of presentation layer, business logic layer, and the control. If you mix them together, it is hard to componentize a system.

Using a well-defined template for presentation layer is a way to componentize the presentation layer. A well-defined template will make the change of the look-and-feel of the web-based system much easier. It is better to create a well-defined template for the web presentation at the beginning and have every developer stick to the template.

Principles for Construction Phase

Principle #7: Group learning and peer review among developers

This is very important in an organization that has very limited developer resources and financial resources. Nowadays programming is not simply working in a centralized environment with a programming language. Programmers have to know the programming language, their development environment usually an IDE, the server environment, and the pros and cons of a distributed computing context. The programming language itself is much more complex than its precedence. It is hard for a programmer to master every aspects of the programming environment; therefore sharing is important. Group learning and peer review led by an expert are the effective ways to share.

This principle conforms to the characteristics of "dependent on good people" (Baskerville & Pries-Heje 2004). Vidgen (2002) pointed out that an able associate who is quick to learn was an essential prerequisite to the success of the project. Group learning and peer review leverage the "good" people in the project team and help to produce higher quality codes and make others educated in new technologies.

Principle #8: Show system functions with quick protocol

In a non-IT business unit, usually the requirements for a system function are very loosely defined. They cannot have a clear vision of what a system should look like until they see the first functional version of the system. It is not useful to apply rigid software development lifecycle in higher educational organizations where business units do not have staff who can serve as domain expert, someone who knows the business and knows how to translate the business need into a technical requirement. It is almost entirely dependent on the technical personnel who must understand the business or at least partially know the business to translate the business needs into a technical requirement and build the system that provides the functions to meet the requirement and modify it based on the feedbacks of the first semi-production-level protocol. It is important that the protocol is almost ready for production and requires little effort to make it a ready for production product since there are quick needs of time-to-market and limited resources in the organization. In a university system, the IT department does not have the luxury to build a quick protocol to show the working functions and then build the application from the scratch again, architecture-wise and coding-wise after the functions are accepted by the user community. In a web environment, it is probably not necessary either. Therefore, it is very important to lay down a solid foundation for all web-based applications in your organization up front. This leads to our principle #5.

Principles in Transition Phase

Principle #9: Prepare users for the new system with parallel run

This is probably not a problem for a system that is developed for users who have never used the old version of services (in this case, manual version of the services), but there may be a slight resistance to using the new system for those who are used to the old way in obtaining the services. It might be better for the two versions of the systems to run parallel for a while before everyone is comfortable with the new system although parallel systems may mean extra time, money, and resources, they can create a harmonious user community and no one will be in panic when switching to the new system. There are many ways to lure users to the new system, one of which is shown in the principle #3: show real time data. Other ways to lure user to the new system include, but not limited to, making important and critical services available in the new system first, that will make users to have incentives to use the new system; therefore prioritizing services is an important aspect in the new system development. The process of prioritizing services will be definitely governed by business units via domain experts in those business units.

Principle #10 Integrated support system should be part of the system

Although we tried to make the services as intuitive as possible, we still received inquiries on how to use the system. Some users just do not have the patience to read the instruction. An integrated support system is an important part for the system to be effectively used and for the total satisfaction of the end users. Without a good support system that combines human support and unintended automated self-support, the system will be resisted at least among some percentage of users and the complaints from those users will tarnish the IT department's reputation among the upper level management in the university. The bad image of an IT department will definitely influence the IT budget allocation. Although support functions are not the core functions of an IT system, they can influence the success or failure rating of an IT system and they should be taken into serious consideration when designing the system.

Conclusion

In this section, we first discuss the future studies from the guidelines of design science. Then, we assess the implications of the design principles.

Future Studies

One important aspect in design science is evaluating design theory. Evaluation is a crucial component of the research process (Hevner et al. 2004). The evaluation phase provides essential feedback to the construction phase as well as to the quality of the design process and the design product under development. Future studies are needed to complete this study. Future studies include comparing the design theory in this study with other design theories for other systems, such as decision support systems, expert systems, and transaction processing systems. We are also considering the use of a focus group of developers to evaluate the design process and the design product.

Implications

For practitioners, a web-based system for self-services has key advantages over human-attended manual system. It can increase the accessibility and availability of the provided services, decrease the costs of the organization, and increase the satisfaction toward the organization. From this case of development and deployment of WAS system, we can positively report that the WAS system is a great contribution to the university's effectiveness and efficiency and definitely has increased the students' satisfaction with the administrative processes for their routing campus life.

For researchers, although web-based systems have been everywhere, not many researches have been done in term of IT design theory. From this case study, we identified some effective development principles for a WAS system in the specific development and deployment environment. We hope that more studies will come and more principles will be identified for WAS systems so that more generic development and design principles can emerge from these studies by synthesizing principles from all the studies.

References

Bajaj, A., D. Batra, A. Hevner, J. Parsons, and K. Siau "Systems Analysis and Design: Should We Be Researching What We Teach?" Communications of the AIS (15) 2005, pp. 478-493.

Baskerville, R. and Pries-Heje, J. "Short Cycle Time Systems Development: A Comparative Study," *Information Systems Journal* (14:4), July 2004, pp. 237-264.

Baskerville, R. L. and Wood-Harper, A. T. "A Critical Perspective on Action Research as a Method for Information Systems Research," *Journal of Information Technology*, (11), 1996, pp. 235-246.

Chau, M., Zeng, D., Chen, H., Huang, M., and Hendriawan, D. "Design and Evaluation of a Multi-Agent Collaborative Web Mining System," Decision Support System (35:1), April 2003, pp. 167-183.

Davison, R.M. "An Action Research Perspective of Group Support Systems: How to Improve Meetings in Hong Kong," Doctoral dissertation of City University of Hong Kong, 1998.

Gregor, S. "The Nature of Theory in Information Systems," *MIS Quarterly* (30:3), September 2006, pp.611-642

Hevner, A.R., March, S.T., and Park, J. "Design Science in Information Systems Research," *MIS Quarterly* (28:1), March, 2004, pp. 75-105.

Janicki, T. and Liegle, J.O., "Development and Evaluation of a Framework for Creating Web-Based Learning Modules: A Pedagogical and Systems Perspective," *Journal of Asynchronous Learning Networks* (5:2), September 2001, pp. 58-84.

Markus, M.L., Majchrzak, A., and Gasser, L. "A Design Theory for Systems that Support Emergent Knowledge Processes," *MIS Quarterly* (26:3), September, 2002, pp. 179-212.

Nielsen, J. "Designing Web Usability: the Practice of Simplicity," New Riders Publishing, Indianapolis, 2000.

Norman, D.A. "Psychology of Everyday Things," Basic Book: New York, 1998.

Vidgen, R. "Constructing a Web Information System Development Methodology," *Information Systems Journal* (12:3), July 2002, pp. 247-261.

Walls, J.G., Widmeyer, G.R., and EL Sawy, O.A. "Building an Information System Design Theory for Vigilant EIS," *Information Systems Research* (3:1), March 1992, pp. 36-59.

Whitley E.A., "Method-ism in Practice: Investigating the Relationship between Method and Understanding in Web Page Design*," 19th International Conference on Information Systems* (13:16), December 1998, pp. 68-75.