Chapter 14: Wireless Practical Case Study

This chapter includes a continuation of Case Study 13−1 and discusses the addition of wireless components to the airline reservation system.

Continuation of Case Study 13−1

Planning for IT project management covers coordination of management activities, phases, schedule, timeline, and staffing; delegation of duties and responsibilities; exploration of modern technology; and coordination of system software efforts in an organization. The plan focuses on integration of system hardware and software engineering and reengineering goals and principles throughout an organization. The plan also includes adoption of a software reuse approach to system development and maintenance throughout the life cycle. Changing old habits of software development via stovepipe practices is difficult. Practice of modern system development requires time to change culture and mindset. For this case study, the IT project manager covers the following steps:

- Requirements analysis
- Design
- Development of a logical model for the application
- Development of an acceptable user response time when submitting a request to the system
- Operational requirements
- Backup plan
- Definition of practitioners roles
- Life−cycle phases

Requirements Analysis

Once users, customers, and stakeholders agree on requirements, the project manager starts planning. He or she assembles resources and defines system design; prepares a schedule; and defines a phase for implementation, testing, and product delivery to the customers satisfaction.

Design

Design of the user interface for a wireless device user requires more ingenuity than that of a wireline interface because a view screen is limited to possibly four lines. As with a wireline application, the goal is to make the interface user−friendly and build navigation that is intuitive. The user should be able to focus on his or her purpose for using the device and not on the tool itself. The device should be transparent to the user, and all screens should be consistent in look and feel. The fewer the operations required by the user, the more the application will be used.

By developing user profiles, the application can be personalized so that the users experience of the application is focused on content in his or her area of interest. The profile can be specific to the application, in this case an airline reservation system application that focuses on both leisure and business travelers. Since the target audience is generally over 18 years of age, the profile information for this target audience will be different than that of an application that focuses on teenagers. Characteristics that differentiate an audience can also help define the architecture of backend servers and processes. Profiling a user for a wireless device is more challenging than for a wireline device because the design may not allow for the use of cookies and most wireless devices have a limited cache size.

Documentation of user activities into use cases and categorization of them (e.g., critical for application success, nice to have) allows the project team to build an application that personalizes the users experience. Critical functionality consists of activities that are required and that all users must perform. Within a use case, path activities are detailed into a main path and alternate paths, including exceptions and errors. Within each categorization, activities can be sorted according to user frequency. Based on the use cases, a use case diagram can be created to reflect a hierarchic tree of activities based on frequency. This exercise allows for a design in which the most often used functionalities are either always available or can be navigated with the least keystrokes. Bad designs are discovered during user testing.

Development of a Logical Model for the Application

Wireframes help maintain a consistent user interface. Since the display size is different from that of phones, personal digital assistants (PDAs), and two−way pagers, and hard coded keys are not universal on all mobile phones, developers must create different interfaces for each device and major brands.

For example, the back, up, and down keys are currently hard coded on mobile phones, and right and left keys may not exist. Other constraints that affect design in mobile phones consists of varying fonts on view screens, lack of graphics support, and soft−keys that may limit text size to five characters. To assist software developers in designing user interfaces, many manufacturers have made emulators available on their Internet sites. A wireless user interface consists of a deck and cards. A card consists of content, and a deck consists of one or more cards.

Definition of Acceptable User Response Time When Submitting a Request to the System

For this case study, the system that is to be made web enabled is a client/server application; thus the airline does not have a baseline and there is no hard response time requirement. Neither of the two airline upgrade projects (profiling or direct connect) has an explicit, quantitative requirement for end−user response time. However, the airline would like to maintain at least the current acceptable level of web site performance. The airline has indicated that an acceptable level of variation from the current performance is approximately 10%. Higher response times than that should be cause for project level consideration of detailed analysis of the bandwidth bottlenecks and development of performance optimization strategies. To provide a basis for measurement of response, a set of web pages, whose functionality does not radically change between now and the upgraded web site, needs to be defined as the common metric. Of the three types of end−user accessible web pages that are being developed (home pages, registration pages, and reservation pages), the latter two are completely new functions and are not available for baseline data.

Unfortunately, there is currently no hard data on response time performance of the airline site. The only partial measurement is data on the global navigation bar, which show an 8−second load on a 28.8 bps dial−up line; however, that is only one section of any page. The airline is currently launching efforts to collect such data and measure the current response times in milliseconds from the proposed Internet service provider (ISP). The point in gathering these data is to provide a consistent and measurable test entry point. This measurement should be completed before the systems design begins.

Operational Requirements

Several operational requirements must be met, including the following:

- ***Availability.*** The availability of the airline generally and the software applications produced depend principally on the systems availability provided by the production system housed at the ISP. That service level of availability is 99.5% as defined in the airlines contract with the ISP.

- ***Redundancy.*** To ensure the availability requirements of the system, the design of the applications must accommodate redundancy through multiserver operation on each operational tier (i.e., web, application, and data). This should include the ability of the system to automatically detect and redirect operations to backup servers. Currently this includes the following:

    - ♦ Load−balanced web servers
    - ♦ Cold backup Direct Connect XML session server (compatibly configured servers with application software)
    - ♦ The Forte software run−time environment provides its own failover redundancy
    - ♦ The airlines telecommunications organization planning redundant dedicated lines
    - ♦ Dual T−1s initially from the production environment to the airlines data center
    - ♦ Later design decisions that may introduce additional elements, such as clustered DBMS servers, further load balancing, or load−sharing application servers, these latter elements are determined during the course of detailed design and implementation

Backup Plan

The airline has developed a backup plan to address issues of catastrophic failure of the system, the airlines Y2K Contingency and Recovery Plan. This backup plan ensures that any user−related data are backed up and archived on a regular basis, the backup operation will not interfere with the normal operation of the profiling system.

Definition of Practitioners Roles

Technical architects are responsible for the following:

- Assessing the project needs and client environment
- Providing time, complexity, and effort estimates
- Recommending, designing, and developing the system architecture
- Designing and potentially creating the application framework/foundation
- Developing standards and strategies for coding and foundation testing
- Understanding and implementing the technical architecture
- Assessing team resources
- Allocating, planning, and managing technical tasks
- Making day−to−day decisions
- Guiding the team

Technical Architect Roles

- Architect owner
- Technology expert, guru, and analyst
- Technology salesperson
- Application leader
- Technical thought leader
- Team member
- Coach and guide

Information Architect Roles

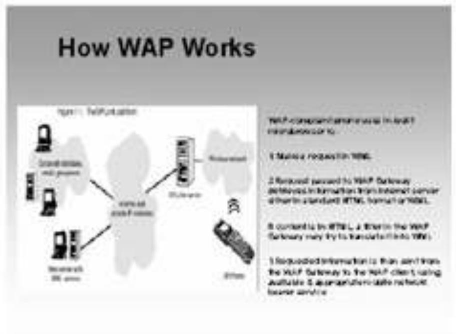
The information architect is responsible for managing information, ranging from transfer of content from a server to user interface. The users experience will be heavily influenced by how this architect designs the system. The information architect must be able to do the following:

- Write test plans, conduct usability testing, and compile results.
- Conduct usability testing and compile results.
- Understand the clients business and content, then produce a site flow document as a solution and present the solution to the client.
- Produce wireframe diagrams on projects with heavy information design requirements.
- Demonstrate a high level of communication and collaboration with clients, the design team, and other creative practices.
- Conduct the information design portion of workout sessions with clients.
- Act as team leader and mentor for other information architects.
- Understand all aspects of the creative process.
- Contribute to business development.
- Assist creative producers with management of the clients expectations.

Life−Cycle Phases

Development Environment

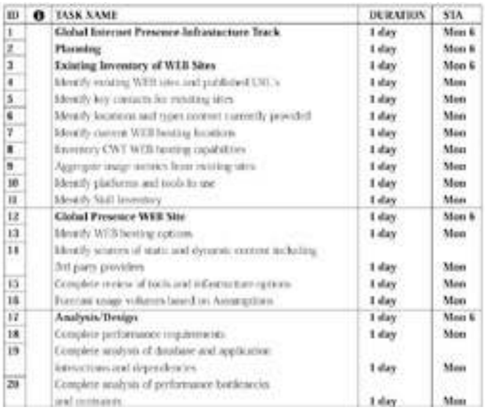
The purpose of the development environment is to demonstrate end−to−end functionality on the web site (Figure 14−1). The hardware and software infrastructure is sufficient to fully exercise all aspects of the target technology (Figure 14−2). However, it is not sufficient to demonstrate full−load stress testing or failover testing. A full−regression testing configuration is implemented and replicated in each subsequent environment. Software engineers writing the flight reservation system code will do all of their development in this environment and test their own modules. Thus the environment does not have to match a real−life scenario in which objects would be distributed for load balance and failover. Since the focus is to offer software developers an efficient development environment, a project manager will be responsible for obtaining large development computers and empowering as much system administration as the IT department will render and as the project development team will accept.



How WAP Works

[illegible]

Figure 14−1: How WAP works



ID	ⓘ	TASK NAME	DURATION	STA
1		**Global Internet Presence Infrastructure Track**	1 day	Mon 6
2		**Planning**	1 day	Mon 6
3		**Existing Inventory of WEB Sites**	1 day	Mon 6
4		Identify existing WEB sites and published URL's	1 day	Mon
5		Identify key contacts for existing sites	1 day	Mon
6		Identify locations and types content currently provided	1 day	Mon
7		Identify current WEB hosting locations	1 day	Mon
8		Inventory CWT WEB hosting capabilities	1 day	Mon
9		Aggregate usage metrics from existing sites	1 day	Mon
10		Identify platforms and tools to use	1 day	Mon
11		Identify Skill inventory	1 day	Mon
12		**Global Presence WEB Site**	1 day	Mon 6
13		Identify WEB hosting options	1 day	Mon
14		Identify sources of static and dynamic content including 3rd party providers	1 day	Mon
15		Complete review of tools and infrastructure options	1 day	Mon
16		Forecast usage volumes based on Assumptions	1 day	Mon
17		**Analysis/Design**	1 day	Mon 6
18		Complete performance requirements	1 day	Mon
19		Complete analysis of database and application interactions and dependencies	1 day	Mon
20		Complete analysis of performance bottlenecks and constraints	1 day	Mon

Figure 14−2: Wireless tasks

Testing Environment

The purpose of the testing environment is to validate functionality during formal tests and perform stress testing on the hardware platform that is similar to the production environment. Full−load testing and failover testing occur during this phase.

Integration Environment

The integration environment is permanent and is used to merge the airline companys upgrade project code and data with those of other development and maintenance projects.

Staging Environment

The staging environment is either located at the ISP (with connections back to the production Forte and mainframe systems) or at the companys data center, depending on whether the IT department will accept responsibility by installing the proper infrastructure for meeting the availability requirements (first−line support). This is a permanent environment to ensure that all software components are properly configured and are ready to be placed into a production environment. This provides a brief, final sanity check at the ISP facility. No significant testing or corrections are expected to occur within this environment.

Production Environment

The production environment is located at either the companys ISP or data center. The application is located at the data center and is connected by dedicated lines. This is a permanent environment, and the current configuration must be upgraded to meet the new requirements.

With the requirements and architecture described in this chapter, a project plan can be assembled to reflect the development effort required.

Section IV: Building an IT System

Chapter List

Requirements are the foundation of an IT project. Failure to understand the requirements in the beginning results in an incorrect system and probably delays in delivery.