1. Activity #1

**Purpose of this exercise:** Establish the high-level requirements for a software product

**Problem:** You are to determine the user requirements for a **web phone-mail product**. The primary purpose of this product is to give phone-mail users (e.g., faculty and staff) the ability to access the functionality of the phone-mail system from a web page. In general, the product should enable users of the phone-mail system to do their usual phone-mail activities via a web page. You will need to determine what are those activities, in the form of product requirements.

**Customer:** You will need to assume the role of a hypothetical customer to identify the **important user-level requirements**. You will also need to assume the roles of engineer and user in order to develop the full requirement specification.

**Solution Activity #1**

1. Requirements. Because the web phone mail system is to provide the basic functionality of the existing system, it should probably include most of the following:

- Users will be able to review the set of messages in their mailbox
- Access to the user's mailbox will be password protected
- The messages that are new messages will be identified.
- Users can delete and save messages in the mailbox
- Users can forward a message to another user.
- Users can listen to a message

Some additional capabilities like the following are plausible. These are probably of lower priority.

- Date and time and message length will be shown for each message in the mailbox
- Users can pause, fast forward, rewind, and stop the messages that are playing.
• Users can reply to a message if they have a microphone attached and they are on campus.
• Users can change their password.
• Users can change their answering message (if they have a mike and are on campus).

Some of the nonfunctional requirements might include:

• (Performance) Response time should be interactive (< 6 seconds delay to play a message) if users are on campus.
• (Security) Communication should be secure (e.g. with encryption)
• (Reliability) e.g. single user failures will not crash the entire system
• (Maintainability) - Design will include provisions for periodic software updates
• (Security) - Unattended sessions will require re-login
• (Usability) - Users will be able to access the mailbox in <5 mouse clicks
• (Usability) - Users can obtain context-sensitive help from any panel
• (Usability) - Users will be able to undo any destructive action
• (Browser) - System will work on Internet Explorer 5+ and Netscape 4+
• (Platform) - System will work with Windows 2000, Windows XP, Mac OS X
• (Performance) - System will provide interactive playback on 500Mhz Pentium II or better systems

2. Requirement evaluation. All the above requirements satisfy the 8 criteria: Understandable, Verifiable, and Independent of implementation, Consistent, Complete, Unambiguous, Realistic, and Necessary.

3. Prioritization. The first six functional requirements are high priority. All of the above non-functional requirements are of high priority; none of them are optional.

4. Risks: There is a significant risk that the programmers will not be able to produce the product in time, especially if they are not experienced. Other likely risks include the likelihood that the customer will change the requirements, or that some of the staff will leave, or that the funding will be cut.

5. System requirements of users being able to review their mailbox contents.

System requirements: Review mailbox contents.

When user requests "view mailbox" the mailbox contents will be transparently encrypted and transmitted to client browser.

• A locking mechanism will guarantee that only one user session can update the mailbox at one time.
• All user accessible fields of the phone-mail database will be displayed and selectable in an html form
• Users will select action to perform by mouse click.
• After action is selected, user input will be validated for correctness before converting to a proposed database update.
• User display will be updated displaying proposed changes before allowing more user input actions.
• Database updates will be finalized when user logs out.

Solution Activity #2

1. Determine actors of the system

Primary Actors:
1. Web phone-mail user
2. Telephone system
3. Web server
4. Web phone-mail system administrator

Secondary actors:
1. Client web browser
2. Web server administrator
3. Telephone system administrator or operator
4. Existing campus data network system (Voice over IP system)

2. Identify Use Cases (following include answers to questions 3 and 6):

a. Authenticate user: User types in his/her id and password to start a web phone-mail session

- Primary Actor: End-user (faculty and staff)
- Secondary actors: Existing telephone system, web server, web browser client
- Goal: User is authenticated and authorized to use the system
- Precondition:
  - User account has been defined in the system
  - User is connected to phone-mail system using a web browser.
- Post condition:
  - User session is established
  - User is able to view mailbox summary in browser

b. Review contents of mailbox: Enable user to view messages and perform action on selected message.

- Primary Actor: End-user (faculty and staff)
- Goal: User is able to view messages and perform operations on any of the messages in the mailbox.
- Precondition: User is connected to phone-mail system using a web browser
- Post condition: User performed specific action on available system functionality.
c. Play a message: Enable user to listen to a selected message and to re-play, fast-forward, rewind, pause, and stop the message
   - Primary Actor: End-user (faculty and staff)
   - Goal: User is able to retrieve and hear contents of message
   - Precondition:
     - User is connected to phone-mail system using a web browser
     - User has a message to play
   - Post condition: User is able to hear contents of message.

d. Forward a message to another user: Enable user to send a copy of a message to another phone-mail user
   - Primary Actor: End-user (faculty and staff)
   - Goal: User sends a message from his/her mailbox to another phone mail user.
   - Precondition:
     - User is connected to phone-mail system using a web browser
     - Receiving user must be have been defined in the phone-mail system
     - User has a message to play

e. Establish a new user account: A new user is defined and an account in the web phone-mail system is established.
   - Primary actor: Web phone-mail system administrator
   - Secondary actors: End-user, existing phone-mail system, web server
   - Goal: Create new web phone-mail account
   - Precondition: Web phone-mail system administrator has logged in with credentials to identify administrative privileges to the web phone-mail system.
   - Post conditions:
     - An end-user account is established
     - End user is provided with account information to use web phone-mail
4. Use Case diagram (following include answers to questions 4 and 5):

7. Two Scenarios:

A. Establish a new user account.
Goal is for a new user to be defined for an account in the web phone-mail system.

1. The Web phone-mail system administrator logs into server with administrative and is authenticated with administrative privileges.
2. The Web phone-mail system displays a menu of administrative functions
3. The administrator selects "Create new user account"
4. The administrator is presented with a user creation form to be filled with required information.
5. The administrator obtains information from user.
6. The administrator supplies enters user information into the system and clicks submit.
7. The Web phone-mail system validates submitted entries
8. The administrator requests chooses the option to create a new account
9. The Web phone-mail system displays an option to select if account will be web enabled
10. The administrator chooses "web-enabled" by clicking on that option.
11. The Web phone-mail system confirms selection by displaying newly created account with option to email information to new user.
12. The administrator chooses “Notify user by email” and then the system goes back to the main menu.

B. The "Review contents of mailbox" use case scenario is:
1. The end user logs into Web phone-mail system
2. The end user is presented with a list of all messages in his/her inbox
3. The inbox list shows the messages that have been heard/reviewed, and those that are new.
4. The list is sorted by time received. New messages are highlighted.
5. The end user chooses a message from the list.
6. The end user views each message with caller identifier, message length, and date and time received.
7. Selected message is displayed with detailed information. Possible action buttons corresponding to individual system functions are enabled.
8. User clicks on an action to perform (delete, forward, play, save, quit)
9. Specified action is performed with option to return to main selection list.
10. The user can is sent back to step 2 with option to perform available functions on other messages
Solution Activity #3

Your assignment is the following:
1. Draw a UML Sequence Diagram to illustrate the above use case. The classes in this diagram should include "InventoryManager" (The inventory management class that has been provided for you), "SessionManager" (The class that reads and writes console messages), and "BookOrder", a class that contains the information associated with this order (the book title, the ISBN, the number to be shipped, the number back-ordered, the shipping name and address, and the total charge). Add any additional classes you think appropriate.
2. Draw a UML Class Diagram that illustrates the classes and their associations.
Solution Activity #4

Design Review by answering the following questions and determining improvements:

1. Does the design meet the Requirements? State specifically which (if any) requirements are not met, and propose corrective action. Note if any of the Use Cases will not function as expected.
2. Does the design include any unnecessary features? Identify any capabilities that can be eliminated from the design while still meeting the requirements.
3. Is the design adequately specified? Note any areas where the design has not been sufficiently explained to perform an evaluation. Note that this is a high-level design review; the design is not expected to specify all low-level details.
4. Are there significant design flaws? Identify any parts of the design that are not likely to function as required. For each such flaw, propose changes that will address the problem.
5. Is the proposed design well organized, according to object-oriented principles? Identify any organizational problems and specify how they should be improved.
6. Is the proposed design likely to be implementable? Note any areas of the design that will be difficult to implement in a timely fashion, and propose improvements that will resolve these difficulties.
7. Usability: Will the proposed design enable users to easily perform all needed functionality? Identify any problem areas and propose improvements.

Requirements from Design Review

1. Users will be able to specify travel dates and times, start and end airports, number of passengers, and will be provided a list of available outbound flight sequences and return trips that meet their requirements.
2. A User will be able to choose a departing and returning flight sequence from the list of available flight sequences, and the number of passengers. If the flights can be reserved, the system will show a full itinerary and specify the price. The flights will be held for sufficient time (1 hour) for the users to book the reservation.
3. If the flight is no longer available, the user will be so informed, and asked to make another choice.
4. If the user asks to make a reservation, the user is presented a page for submitting credit card information.
5. Before credit card information is submitted, user is informed that the flight will be booked and they will not be able to obtain a refund if they proceed.
6. If the credit card submission is validated, a reservation is booked, and a confirmation number is presented to the customer.
7. At any time prior to booking the reservation, the customer can propose changes to the proposed travel. Existing holds on flights will be removed if the customer changes the itinerary.

Flight Booking System API calls

- Find all flights from airport A to airport B between “time A” and “time B” on specified date (A and B are 3-letter airport codes). Specifies number of seats, price.
- Determine (city, state, country) from airport code
• Determine airport code from (city, state, country)
• Put an N-minute hold on S seats on a specified flight number on a specified date. 0<N<100, space must be available. (Returns a code)
• Release a hold (for specified code)
• Book a reservation for 1 person on 1 flight. Must specify flight hold code, passenger name, authorized travel agency that is billed for flight. Returns: Confirmation no.
• Given airport A and airport B, identify all airports C that are acceptable stopovers en route from A to B
• Associate frequent flyer number with confirmation number. Name and frequent flyer no must match or will be rejected.
• Set passenger preferences – aisle or window, meal, etc. for a confirmed passenger reservation

Credit Card billing API
• Specify name, billing address, credit card no., expiration date, charge amount, creditor
• If successful, transfers funds to creditor account, plus transaction ID.
• If unsuccessful returns failure code (e.g. insufficient credit, invalid number, etc.)

Class Diagram
IMPROVEMENTS

(Answer to #5: Is the proposed design well organized, according to object-oriented principles? Identify any organizational problems and specify how they should be improved.)

The designed flight class was reviewed and revised to capture attributes and operations as required by the system.
**Answer to #6. Is the proposed design likely to be implementable?**

Yes the design is implementable, though there will be difficulties in timely implementation because of dependencies on other systems to which the flight reservation system will interact. These difficulties are:

- The card charge billing systems, flight itinerary data and information from various airline carriers and;
- Scheduling coordination and resolution based on the departing airport and/or destination airport, plus;
- Changing rules and regulations to which the airline industries have to abide and subsequently be captured as business rules of the system.

If the system were to be implemented, a typical flight-booking scenario would be as follows:

1. A passenger requests for flight information to a planned destination
2. System searches the flight database for matching flights based on user requested flight criteria
3. Flight booking system displays selected flights result set
4. User has option to reserve a selected flight and system holds selected flight records
5. System enters a “New” flight itinerary for selected flight
6. Flight booking system enters selected flight to add to proposed itinerary list
7. Flight booking system displays price schedule of selected flight to the user
8. System displays proposed Itinerary
9. User accepts proposed flight itinerary
10. System displays charge form
11. Flight booking system creates a new billing record
12. Flight booking system submits itinerary information to billing system
13. Flight booking system sends charge information to billing system
14. The billing system applies the charge for the selected flight
15. Flight booking system updates the confirmed and charged flight record
16. The billing system displays flight booking confirmation to the user
17. User has option to end session or return to step 1 for more flights.

**Answer to #7. Usability: Will the proposed design enable users to easily perform all needed functionality? Identify any problem areas and propose improvements.**

Yes, the iterated and reviewed design will enable the users to perform the functionalities and requirements as specified above. The design was based on addressing the defined requirements. Most likely, there will be problem areas and room for improvement during implementation; however, the design will be flexible enough to handle changes not captured in the original design.