LUCID STAGE 1: ENVISION

- Develop “high concept” for product
- Produce UI roadmap, that defines concept, constraints, user classes, usability goals
- To create shared vision of product, as described in UI roadmap
- Goals and corresponding LUCID activities (LUCID tasks)
  * Create concise description of product — a “high concept” statement
  * Identify user classes
  * Identify usability goals
  * Identify constraints
  * Identify main functionality (user tasks)
  * Produce UI roadmap

INTRODUCTION TO CALENDAR MANAGEMENT SYSTEM

- Calendar Management System
  - * Simple automated version of a paper calendar
  - * To be used for in-class examples of stages of LUCID
  - * Goal is to learn the development process, not to produce a marketable calendar product
- To give you some concrete examples of how each LUCID activity might progress

HIGH CONCEPT STATEMENT FOR PRODUCT

- High concept statement — brief descriptive summary of product, typically 25 – 50 words
- Mission statement for a product, to help focus product development
- Writing a good high concept statement is not easy and is not done once; highly iterative
- Answer following questions:

  * What is product name?

  * Who are product users?

  * What will product do?

  * What problem will product solve?

HIGH CONCEPT STATEMENT FOR PRODUCT

- A possible high concept statement for Y2K Calendar:

  * Our calendar will have automated support for scheduling appointments, to improve customer satisfaction.

- A better high concept statement:

  * The Y2K Calendar will allow a broad variety of users to schedule and manage appointments. These users can range from professionals using the system to run an office to casual users keeping track of personal information. Automated support will reduce scheduling effort and increase awareness of appointments.

USER CLASSES (OR USER POPULATION)

- Need to take into account human diversity and wide range of situations, including
  - * User knowledge of domain
  - * User knowledge and prior use of computers
  - * Training and application-related experiences
  - * Frequency of use
  - * User goals
- “Know thy user” — and it is *not* you!
  - * Important to have representative user(s) on development team and/or have access to representative user(s)

USER CLASSES (OR USER POPULATION)

- Generic classification of user types
  - * Novice or first-time user: May know application domain but does not know specifics of application
  - * Intermittent user: Uses several systems from time to time; knows application domains but does not remember details of different applications
  - * Frequent user: "Power" user who probably uses application daily and knows both application and task domain very well
- Try to account for each of these user types in interaction design

USER CLASSES

- Example of user classes for Y2K Calendar:
  - * General characteristics
    - Busy people
    - Keep schedule for self and others
    - Professional and personal use
    - Calendar is very small part of job
    - Need ‘transparent’ tool
    - High general skill level, literate
  - * Domain skills
    - Know how to use calendar
  - * Computer skills
    - Broad range
    - At least some typing skills
    - Familiar with GUI/mouse

USER CLASSES

- User classes (continued)
  - * Conclusions
    - Keep it simple
    - Usability as important as functionality (or more)
    - Try to get functionality greater than paper calendar
    - Try to get usability greater than paper calendar
    - Minimize typing
    - Users must learn it quickly
- Caution: Difficult for users to tell developers what they want or need
  - * Important to observe users in their typical work environment

USABILITY GOALS

- Project-specific usability goals identify objectives in terms of usability and design of user interaction
  - * Reflect real use of product in real world
  - * Determine what is important to organization and to users
- Usability evaluation design (later in semester) driven by usability goals
- Determining usability goals in terms of
  - * User classes and workgroups
  - * User task context, special tasks
  - * User errors

USABILITY GOALS

- Errors
  - * How important is it to avoid user errors?
  - * What are consequences of errors?
  - * How will errors be reduced or avoided?

USABILITY GOALS

- User classes and workgroups
  * Is product for novice, intermittent, or frequent users?
  * What domain knowledge do users have?
  * What computer knowledge do users have?
  * Are users discretionary or captive?
  * Are users typically working alone or in a group?
- User tasks
  * What are representative, most common tasks?
  * What are key, mission-critical tasks?

USABILITY GOALS

- Example usability goal for Y2K Calendar:
  * Reduce amount of time for user to perform task of adding recurring appointments (based on current version, either paper calendar or some other automated calendar)
- Can quantify to be even more precise
  * Example: Currently 25 seconds to perform task of adding recurring appointment; reduce to 15 seconds
- Another example usability goal:
  * Reduce number of errors made by users in rescheduling an appointment (based on some current automated calendar)
- Another example usability goal:
  * Increase effectiveness of calendar by helping users avoid missed appointments

HIGH-LEVEL CONSTRAINTS

- Welcome, once again, to the real world!
- Some possible constraints
  - Hardware
  - Environment of use
  - Marketing inputs/requirements
  - Cost and budget
  - Schedule and development time
  - Size and/or weight
  - Integration with existing or other developing systems
  - Security

HIGH-LEVEL CONSTRAINTS

- Hardware
  - * Will hardware be customized or standard?
  - * Will product be used on variety of platforms?
  - * Are specialized devices (i.e., beyond standard keyboard and mouse) needed?
- Environment of use
  - * Will product be used in private or in public locations?
  - * Will surroundings be noisy? dirty? dark? damp?

HIGH-LEVEL CONSTRAINTS

- Cost and budget
  - * What is allowable budget?
  - * What restrictions does this impose on product scope?
- Schedule and development time
  - * What is current time frame?
  - * What restrictions does this impose on product scope?
- Size and/or weight
  - * Will product be on portable or mobile equipment?
- Integration with existing or other developing systems
  - * What other applications does product have to interact with?
  - * Are data across these applications compatible?
- Security
  - * What privacy and/or security issues are there?

HIGH-LEVEL CONSTRAINTS

- Example high-level constraints for Y2K Calendar:
  - * Product will be used in a wide variety of environments, ranging from factory floors to open offices at work to home
  - * Product will run on a wide variety of standard platforms, but most will be personal computers with typical input devices (keyboard, mouse, possibly a touchpad such as on a notebook computer) and no special devices
  - * Current budget is not determined
  - * Current development time is one semester

FUNCTIONALITY/USER TASKS

- Statements that describe highest level of user tasks to be supported by product
  - * Describe what product will do for user
  - * Define broadest possible scope of development effort (narrowing of scope comes later, based on more information)
- IMPORTANT: Express user tasks in user terms, rather than in system terms! For example:
  - * View appointments — what a user does
  - * Display appointments — what a computer does

FUNCTIONALITY/USER TASKS

- Where does knowledge of user tasks come from?
  - * At this point in process, comes from brainstorming with development team
  - * Especially if product is particularly unusual or broad, there should be a domain expert on development team
  - * Later in process, comes from interaction with representative users and/or more domain experts
- Each statement should begin with “User will be able to…”
  - * Be sure to keep statements user-oriented, not system- or computer-oriented

FUNCTIONALITY/USER TASKS

- Example of functional/user task statements for Y2K Calendar:
  - * User will be able to add appointments
  - * User will be able to delete appointments
  - * User will be able to update appointments
  - * User will be able to add recurring appointments efficiently
- Are these all user-oriented terminology?

UI ROADMAP

- UI roadmap communicates product concept, constraints, and initial user classes and tasks to interested parties
  * Roadmap document is simply a compilation of all previous work done in LUCID Stage 1: Envision
  * Includes:
    - Date of current document; change with revisions
    - Names of team members
    - High concept statement
    - Description of user classes
    - Description of usability goals
    - Description of various constraints
    - Description of main high-level functionality (user tasks)