LUCID STAGE 2: ANALYZE

- Assess user needs and produce task requirements
- Refine user classes, develop early scenarios, produce a user task analysis and describe work flow
- Members of development team work with representative users to document work processes and increase specificity of data gathered in Stage 1
- Goals and corresponding LUCID activities (LUCID tasks)
  * Segment and characterize user classes
  * Gather and analyze data for requirements
  * Document work flow/user task analysis (requirements analysis)
  * Develop usage early scenarios
- Keep focus of activities user-centered!

USER CLASS CHARACTERIZATION

- More details of user classes from Stage 1
  * User segment — a user class with similar characteristics and needs
  * Group based on many different dimensions (e.g., job description, location, level of responsibility, computer literacy, application domain knowledge)
- Developers do not know all this information; *must* interact with representative user(s) to discover it

USER CLASS CHARACTERIZATION

- Answer following questions for each class (segment):
  * What is knowledge of computer technology?
  * What is knowledge of application domain?
  * How complex is application content?
  * Are users discretionary or captive?
  * With whom do users interact?
  * What are training needs?
  * What is user culture?
  * How receptive/resistant to product are users?
- Simply answering these questions about initial set of user classes will probably uncover more user classes!

USER CLASS CHARACTERIZATION MATRIX

	User Segment A	User Segment B	User Segment C
Usage frequency			
Computer knowledge			
Subject matter knowledge			
Complexity of content			
Discretionary or captive?			
With whom do they interact?			
Frequency?			
What info is exchanged?			
Training?			
Culture?			
Receptive/ resistant?			

REQUIREMENTS GATHERING

- Important step in planning for interviews with and/or observations of representatives of each user segment
- In Stage 1 “brainstorming” of user task statements (“User will be able to…”) produced general requirements/user tasks
  * Now will clarify and elaborate on those tasks
- Formally, this is *ethnographic* work
  * Interviews and observations of users, to give insight into behavior and organizational context
  * Participating, “overtly or covertly, in people’s daily lives for an extended period of time, watching what happens, listening to what is said, asking questions” [Hammersley & Atkinson 1983, as quoted in Shneiderman, p. 107]
  * UI designers limit interaction to days or even hours, but have to obtain needed data
    - "Quick and dirty ethnography"

REQUIREMENTS GATHERING

- Goal of ethnography for UI designer is to collect information needed to ensure usability of design
- Process for UI ethnography includes
  * Preparation
    - Understand organization's policies and culture
    - Know current system and history
    - Prepare script of questions for interview
    - Obtain permission to observe and/or interview
  * Field study
    - Establish rapport with managers and clients
    - Observe and/or interview users in workplace
    - Collect quantitative and qualitative data
    - Collect artifacts (e.g., paper forms) as available
    - Follow leads from visits, if any
    - Document visit

REQUIREMENTS GATHERING

- Process for UI ethnography (continued) includes
  * Analysis
    - Compile data into numerical, textual form
    - Quantify data when possible, with statistics
    - Interpret data in terms of UI design
    - Refine process (e.g., script) as necessary for subsequent visits
  * Report process and findings
- Seems easy, but it's not!
  * Hidden traps, surprises (e.g., what to wear to interview, different perceptions of managers vs. users, different use of language/technical terms)
- Equally important as data collected: rapport and relationships with users established during process

REQUIREMENTS GATHERING

- *For each user class*, collect information on:
  - * Steps or actions performed
  - * Objects with which they interact
  - * Secondary sources they consult
  - * Messages sent and received (information flow)
  - * Conditional (if-then) situations

REQUIREMENTS GATHERING

- While this is the information you need to gather, you can't just go to a user and start asking "What objects do you interact with?"!
  * Must formulate interview questions that will "tease out" needed information
  * Have questions ("script") prepared in advance, as a starting point for interview
  * Be ready to modify "on the fly"; follow your script but explore and branch out as necessary
  * First interview may not go too well; regroup for next one
  * Script should include a brief introduction that explains to user why you are there

REQUIREMENTS GATHERING

- *Interview/observation time with user is extremely valuable commodity; prepare and use it wisely*
- Logistically, have to plan how to record information gathered from each user
  * Typically take copious, careful notes during interview and observations
  * Use your script (questions) and leave lots of room for taking notes between
  * Have more than one person taking notes when possible
  * May audio tape or video tape under some circumstances
- Results of information gathering (at least in this course) are a user task analysis and usage scenarios

USER TASK ANALYSIS

- Task analysis — what user tasks are supported
  * *Goal of system*: Manage appointments
  * Assumption: Some boundaries set by management, marketing, customer, etc. (e.g., hardware); determination made that product is novel, market not yet saturated
  * Features
    - *Appointment* means information on:
      Date, Time, Place, Appointment description
    - *Manage* means:
      Add new appointment
      Remove existing appointment
      Change existing appointment
    - Plus, need ability to view/display appointments
  * *Follows from user interviews/observations, not just developers' ideas*

USER TASK ANALYSIS

- Task analysis (continued)
  * Some time later, someone thinks of “alarm” idea
    - Do we want to actively inform of appointments?
    - Decision: Yes, very useful
    - Iterate and revise tasks
  * New feature: Active reminder

USER TASK ANALYSIS

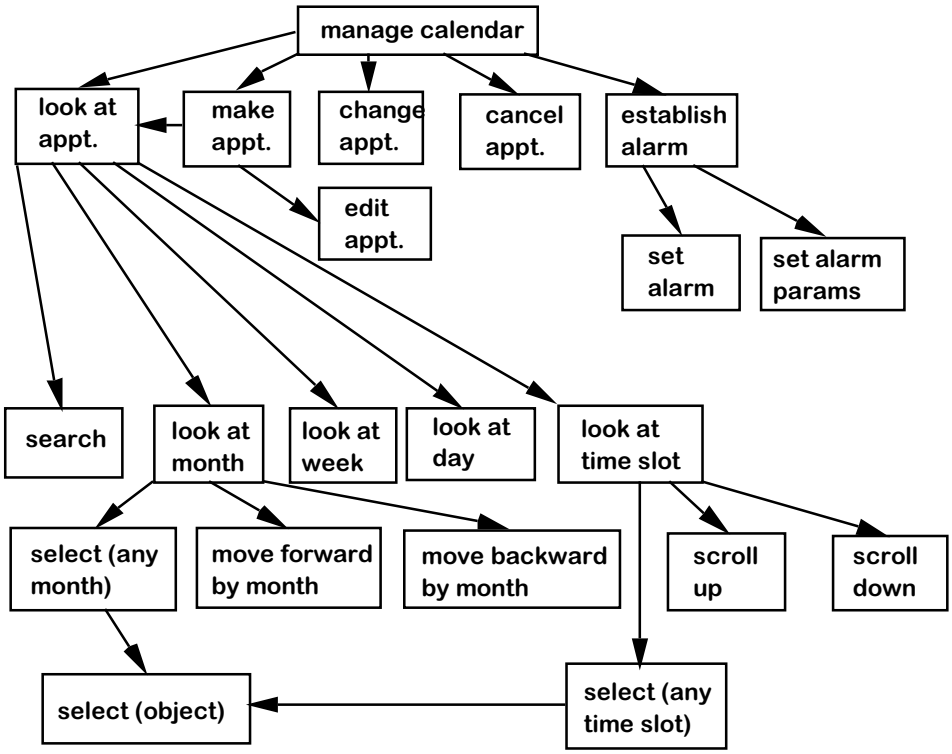
- Task analysis: What tasks will users perform with this system?
  * Tasks are performed by user (e.g., view)
  * Initial list of major sub-tasks
    - Make (add) new appointment
    - Look at (view) existing appointment
    - Change existing appointment
    - Cancel existing appointment
    - Set alarm
    - Look at (view) calendar
  * This is task structure
  * Not timing, precedence, order of task performance, work flow, etc.
  * Only what user *can* do, not *must* do

USER TASK ANALYSIS

- High-level user task actions decomposed into multiple intermediate-level user actions
- Intermediate-level user actions decomposed into multiple low-level (atomic) user actions
  * Choosing atomic actions is difficult
  * Most depend on typing or clicking mouse
- Relative task frequencies helpful in shaping design
  * May want “hot keys” or other shortcuts for very frequent and simple tasks
  * Less frequent or more complex tasks can be less direct (e.g., sequence of menus)

USER TASK ANALYSIS

- Example of possible hierarchical user task structure for Y2K Calendar



- Hierarchical structure is accompanied by brief description of what each box means

USAGE SCENARIOS

- Indicate typical usage threads (“day-in-the-life" of application and users)
  * Can be flow-chart-like and/or textual descriptions
  * Can represent both common and mission-critical situations
  * Can involve different user classes
  * Number of scenarios can be large, if product is large, complex
  * Can include relevant environmental and contextual information

USAGE SCENARIOS

- Example of usage scenario for Y2K Calendar:

  A patient with an existing appointment with Dr. Kevorkian for next Tuesday calls the physician's office. She is unable to keep that appointment, and needs to reschedule it. The scheduler must locate the current appointment, find an open time slot that also is a time the patient is available, and re-enter patient information into the new time slot. While the scheduler is doing this, another phone line is ringing and another patient is standing at the scheduler's desk wanting to make a follow-up appointment with Dr. Kevorkian.

- Note that temporal order of locating current appointment and finding open time slot could be reversed.

  * This raises UI design question: should system force user to perform task as described, or should user be able to perform sub-steps (locating current appointment and finding open time slot) in either order?

  * This decision may be delayed until more detailed design, but it should be noted at this point, so not forgotten

- More on scenarios to come, under Design!