CSC 308 Lecture Notes Week 3
Details of the Requirements Analysis Process

I. This week's material:
   A. Milestone 2 writeup and example.
   B. Requirements document HTML standards.
   C. Conventions for standardized GUIs.
   D. These lecture notes.
   E. Week 3 lab notes, with more on SVN.

II. Some paper UI sketches and other paper notes are OK for Milestone 2, as long as the overall document structure is online in HTML format.
   A. Clearly label all paper materials so they can be referenced from the online requirements document.
      1. Figure number and caption for screen pictures.
      2. Title for other notes.
   B. Make copies if you want to keep the originals.
   C. Deliver to lab or Fisher's office by 5 PM Friday.

III. Recap of general scenario guidelines covered in the Milestone 2 example.
   A. Present scenarios in a *tutorial* style.
      1. Tell an interesting, engaging, and ultimately complete story of system functionality.
      2. Organize scenarios to give the reader a step-by-step presentation of what the system will be like to use.
   B. Start with scenarios showing common activities, without necessarily covering all low-level details.
   C. Separate scenarios based on different categories of end user.
   D. Stylistic recommendations:
      1. Leave mundane details until later, e.g., File, Edit, data entry.
      2. Leave details of error handling until later.

IV. Core steps of the scenario process.
   A. Describe a user action that executes a command.
      1. In the graphical user interfaces (GUIs) with which we are dealing, a user action is performed by some user gesture on the screen.
      2. Physically, a gesture is performed by some combination of mouse and/or keyboard actions.
      3. Most typically, command execution is performed by selecting a menu item, pressing a command button, or typing a keyboard action of one or more keystrokes.
   B. Show the system response to the user action.
      1. In a GUI, the system response typically appears on the screen.
      2. The response can also be shown on some other form of output medium.
      3. In some cases, the system response is not displayed directly to the user, such as the case when a change is made to a an underlying system database that is not visible in any display.
   C. Fully describe the details of the response.
      1. The description is a prose narrative that explains all details of display screens and underlying data.

2. All display screen components are described.
3. All output effects to underlying data are described.

D. If the the system response is an input dialog of some form:
1. Show another picture of it, with representative data values filled in by the user.
2. Fully describe entered values and what they mean.
3. Within an input dialog, if there are non-atomic interactions, cover each using additional screen examples and narrative.
a. Sufficiently simple cases can be covered in the narrative only, without additional screen examples.
b. These include toggles or short lists of alternative selections.
4. If there are input alternatives or options that effect what the system does, show additional versions of filled-in dialog screens to cover all cases.

E. If the system response is output data not requiring user input:
1. One output example is sufficient if it is adequately representative of all functionality.
2. As with input dialogs, if there are output alternatives or options that fundamentally affect the content or format of the presented information, show additional examples and narrative

V. Milestone 6 example excerpt, (to illustrate what you're aiming for, ***not*** what's required for earlier milestones).

A. The accompanying handout illustrates two completed scenarios, circa Milestone 6.

B. Note in particular that this level of completion is not expected for earlier Milestones, but it definitely is for at least some of the Milestone 6 delieverable.

C. These scenarios show what you are working towards in terms of fully completed details.
1. The Calendar Tool may be more complicated in terms of display details and options than your projects.
2. The excerpts were chosen to illustrate some particularly detailed scenarios.

VI. Core steps illustrated in detail in the Milestone 6 example[1]

A. **Describe a user action** -- *see Section 2.2, paragraph 2 on page 9 of the excerpt (starting with "To schedule an appointment, ...").*

B. **Show system response** -- *see Figure 6, page 9.*

C. **Fully describe details of response** -- *see narrative in paragraphs 2 through 6 on pages 9 and 10 (through to and including the paragraph starting "The bottommost ...").*

D. **If response is an input dialog**:
1. **Show another picture of it** -- *see Figure 7 on page 11.*
2. **Fully describe entered values** -- *see the narrative in paragraph 5, page 10 (starting with "After the dialog appears ...").*
3. **If non-atomic interactions, cover them** -- *see Figures 8 -11 on pages 11 and 12 and the narrative in paragraph 6, page 10 (starting "To create a new appointment category ...").*
a. **Sufficiently simple cases in narrative only.**
b. **E.g., toggles or short lists** -- *see the descriptions of* `Security`, `Priority`, *and* `Remind?` *in paragraphs 1-3 on page 10.*
4. **If input alternatives, show additional filled-in dialogs** -- *see Figure 12 on page 13, narrative in paragraphs 8 through 10 on the bottom of page 10 (starting with "The user now proceeds ... ").*

[1] The links in the HTML version of the notes go to HTML version of the Milestone 6 requirements example. The page numbers in the notes refer to the PDF excerpt of same requirements. It's the same stuff, just in two different document formats.

E. **If response is output**:
   1. **One screen sufficient if adequately representative** -- *see the weekly and yearly views in Figures 20 and 23 , on pages 22 and 24, respectively.*
   2. **If alternatives, provide additional examples and narrative** -- *see*
      a. *Figures 13 -17 for the daily view, pages 15-19*
      b. Figures 18 and 19 for the weekly table view, pages 20-21
      c. Figures 21 and 22 for the monthly view, page 23.

VII. Other scenario presentation issues.

A. Ensure complete coverage of functional hierarchy.
   1. Cover all commands and interactions, at all levels of the hierarchy, at least once.
   2. Provide at least two examples of all input dialogs -- one initial version and one filled in by the user; provide additional examples to ensure adequately representative coverage.
   3. Provide at least one example of all outputs, with enough additional examples to ensure adequately representative coverage.

B. Describe specific interface layout details only as they relate to the fundamental display of information.
   1. Do not address purely aesthetic look and feel issues.
   2. E.g., on the bottom of page 14, the details of display layout are described for the daily view based on information content, not anything purely aesthetic (starting with "The day view shows ...").

C. Avoid unnecessarily repetitive presentation.
   1. Refer to pictures or narrative that describe common functionality.
   2. E.g., in the description of weekly view options in the last paragraph at the bottom of page 20, reference is made to the daily view options that are the same for the weekly view, without repeating the daily view narrative. (The paragraph starts "All of the display options available for the day view ...".)

D. Have the scenarios flow by building on information presented earlier.
   1. Refer to examples presented in preceding sections.
   2. Where appropriate, state assumptions about what actions the user has already taken.
   3. E.g., Section 2.3.1.1 paragraph 2 on page 14, and all the figures in Section 2.3.

E. Where necessary, get down to nitty gritty details.
   1. When system functionality is complicated or non-obvious, provide detailed and careful descriptions.
   2. This is the case for the Calendar tool in the discussion of overlapping items for the daily and weekly views, Figures 16-19, pages 18-21.
   3. Use good judgment (and the 7+/-2 rule) to defer very detailed discussions to later sections.
      a. E.g., in paragraph 3 in the middle of page 18, the most complicated details of overlapping items are deferred to Section 2.12.1.2.
      b. In the last paragraph on page 25, important details of changing and deleting scheduled items are deferred to another section, namely 2.4.6.
      c. Without explicit comment in Section 2.2, details of scheduling recurring items are deferred to Section 2.4, entitled "More Individual User Scheduling".

VIII. Interface style issues.

A. Be simple and consistent.
   1. In some cases, a complex interface may be necessary.
   2. In general, however, the interface should be as simple as possible in keeping with the functionality that it provides.
   3. Consistency is always important.

a. The interface should not provide two or more conflicting or redundant ways to perform the same function.
b. When different interfaces are provided for the same function, each interface should provide added value or convenience for the user.
   i. For example, a menu and toolbar interface are frequently provided for access to the same function.
   ii. In this case, the value-added of the toolbar is its iconic form and quicker access than through the menu.

B. Use interface forms that end users can easily understand.
   1. If a user has paper forms or other existing conventions for information display, duplicate the existing forms when they work well for the user.
   2. Try out various styles of interface for the user to see what works best.

C. Provide interface options to allow the user to select among alternate UI forms or display styles.
   1. When two are more interface forms are deemed valuable, provide an option for the user to select which forms are visible at any given time.
   2. For example, an option to show or hide a toolbar in addition to the menubar allows the user to select which form to use.

IX. More on the "interesting and engaging story" told in the scenarios.

A. The purpose of telling an overall story throughout the requirements scenarios is two-fold:
   1. to maintain the reader's interest, so that the reader will stay sufficiently engaged to understand the details of the requirements, and how those details affect the way the software actually will be used;
   2. to provide overall context and continuity, so that examples presented early in the scenarios can be refined in later scenarios.

B. The point is not to entertain the reader in the way a novel or short story does.
   1. Humor should be used sparingly, if at all.
   2. The story should stick to the facts, without flowery prose.

C. The story outline for the Calendar Tool scenarios goes like this:
   1. The user schedules a couple appointments.
   2. The user views her calendar in various ways.
   3. The user schedules some other kinds of items.
   4. The user deals with finer points of scheduling.
   5. An administrative user performs specialized administrative functions.
   6. The user sets calendar options.

D. Story line sketch for CSTutor:
   1. Instructor Builds Simple Lesson
   2. Student Views Simple Lesson
   3. Instructor Builds Advanced Lesson
   4. Student Views Advanced Lesson
   5. Instructors and Students Chat and Interact
   6. Students View their Stats

E. Story line sketch for E-Class:
   1. Instructor builds a simple lecture.
   2. Instructor presents a lecture.
   3. Meanwhile, students view the lecture.

4. Instructor builds an advanced lecture.
5. Students view advanced lecture.
6. Classroom interacts.
7. Instructor performs admin tasks.

F. Story line sketch for the Grader:
1. Instructor downloads and sets up roster.
2. Instructor adds items and students.
3. Instructor views charts and graphs.
4. Instructor adds gradesheet details.
5. Student Views and Predicts Grades.
6. Student views historical information.

G. Story line sketch for the Scheduler:
1. Instructor Sets Up Preferences
2. Admin Creates Simple Schedule
3. Admin Edits Databases
4. Admin Creates More Complicated Schedule
5. Admin Deals with Scheduling Constraints
6. Student Views and Comments on Schedules

H. Story line sketch for the TestTool:
1. Instructor creates a simple test.
2. Instructor edits question database.
3. Instructor creates more complicated test.
4. Student takes test.
5. Instructor grades test.
6. Instructor manages tests and question DB.

X. Concrete data underlying the requirements scenarios.

A. In conjunction with the scenario story line, there should be a consistent set of example data to support the scenarios.
1. These example data must be extensive enough to support all of the scenarios in the requirements.
2. At the same time, the data need be no more expansive than is necessary to support the scenarios.
3. The data exemplify a variety of realistic examples, in conjunction with the overall scenario story line.
4. Typically, no single scenario shows all of the data, but rather scenario-specific excerpts of it.
5. A requirements document appendix can show the complete content of the underlying example data.

B. In the case of large data collections, the scenarios are organized into two broad categories of *data editing* and *data viewing*.
1. Data-editing scenarios present functionality to add, modify, and delete data elements in the collections.
2. Data-viewing scenarios present functionality to search the collections, and show the data in whatever forms are available to the user.
3. For data-editing scenarios, a sufficient number of representative examples are shown, but there is no scenario that shows each element being added individually to a large collection.
4. Rather, following scenarios that show representative data elements being added, the narrative can say something to the effect of *"The user now proceeds to add additional data elements, in the same manner as described in the preceding scenarios."*

5. The subsequent data editing and viewing scenarios then display scenario-specific collection elements.

C. A fundamentally important reason to have consistent underlying data examples is to provide continuity through the scenario story line.

1. Typically, early scenarios show data being created.
2. Subsequent scenarios show the same data being modified, and selected pieces of data being deleted.
3. After that, scenarios present viewing functionality for the same data that have been created in the preceding editing scenarios.
4. In some cases, viewing scenarios may come first, before editing details.
   a. In such cases, the narrative says something to the effect of *"The following scenarios assume that the user as created some data ... . Full details of creation are covered in the scenarios of Section X."*
   b. Continuity is maintained by having the subsequent editing scenarios use example data that appeared in the preceding viewing scenarios.

XI. Data examples for the 308 projects.

A. Discussed below are the major data examples for this year's 308 projects.

1. These are example data that provide continuity among the scenarios that present the major functionality of the tools.
2. The discussion does not cover data examples for all tool functionality; the focus is on the major stuff.

B. In the Calendar Tool requirements, there are consistent underlying calendar examples for a number of users, and consistent examples for each of the databases.

1. The main example used in most of the scenarios is the work calendar for a single user.
   a. It has a sufficient number of scheduled items to support the scenarios for scheduling and viewing an individual user's calendar.
   b. It provides continuity across all of these scenarios.
2. There are also smaller examples for the same user's personal and off-campus calendars; these examples support scenarios that illustrate multi-calendar functionality.
3. To support scenarios for meeting scheduling and viewing other-users' calendars, there are underlying example calendars for a number of other users.
   a. All but one of these calendars have only a few items each, to support the scenarios for meeting-scheduling functionality.
   b. One of the calendars has enough items to show full details of viewing other user's calendars.
4. There is one full example for each of the three databases.
   a. These have a sufficient number of entries to support database viewing and editing operations.
   b. They provide continuity across the scenarios for regular user and administrative user functionality.
5. Finally, there is an appendix that shows the complete content of the example calendars and databases.

C. Underlying data for CSTutor:

1. You need example lessons and a student account database.
2. A single main lesson can be used in most of the scenarios.
   a. It must have a sufficient number of pages to support scenarios for lesson authoring and viewing, including quizzing.
   b. A few additional smaller lessons can be used to support the scenarios for the inter-lesson navigation, and multiple lesson scoring for students.
3. You can provide just the shells for a number of additional lessons, to fully illustrate the lesson database and its browsing features.
4. You also need an example student user database, that has enough student entries to support scenarios on student/instructor interaction in chat sessions, and overall student statistics.

D. Underlying data for EClass:
   1. You need sample lectures and student roster.
   2. A single main lecture can be used in almost all of the scenarios.
      a. It must have a sufficient number of topics and slides to support scenarios on lecture preparation, presentation, viewing.
      b. In particular, the lecture must varying topic depths to thoroughly illustrate the various navigation features, including expand and collapse.
   3. One or two additional lectures can be provided to illustrate how lectures are stored and transported.
   4. You also need a sample student roster, with a sufficient number of students to illustrate all of the roster-related functionality.

E. Underlying data for the Grader:
   1. You need example gradesheets and an example SIS student roster.
   2. A single main gradesheet example can be used in most of the scenarios.
      a. It must have a sufficient number of student rows and graded-item columns to support the scenarios for row/column operations, graphic display, and student what-if scenarios.
      b. It provides continuity across all of these scenarios.
   3. Additional smaller gradesheet examples can be used in the scenarios that show grade trends, and storage of gradesheets on the server.

F. Underlying data for the Scheduler:
   1. You need example generated schedules, and example databases.
   2. A main schedule example schedule must have a sufficient number of scheduled classes to illustrate generating and editing a good schedule.
   3. Additional smaller example schedules can be used in the scenarios that cover specialized scheduling constraints and preferences.
   4. The example databases should have a sufficient number of entries to support all forms of scheduling; most likely, a single example for each database should be sufficient.

G. Underlying data for the TestTool:
   1. You need example tests question database(s).
   2. A single main test example can be used in most of the scenarios.
      a. It must have a sufficient number of questions to support the scenarios for test generation, test taking, and test grading
      b. It provides continuity across all of these scenarios.
   3. Additional smaller test examples can be used in the scenarios that illustrate details of test generation.
   4. A single main question database example must have a sufficient number of questions to support test generation and database viewing scenarios.
   5. If your team chooses to work on functionality for question database sharing, then you'll need some additional (smaller) database examples for these scenarios.

XII. Screen maps.

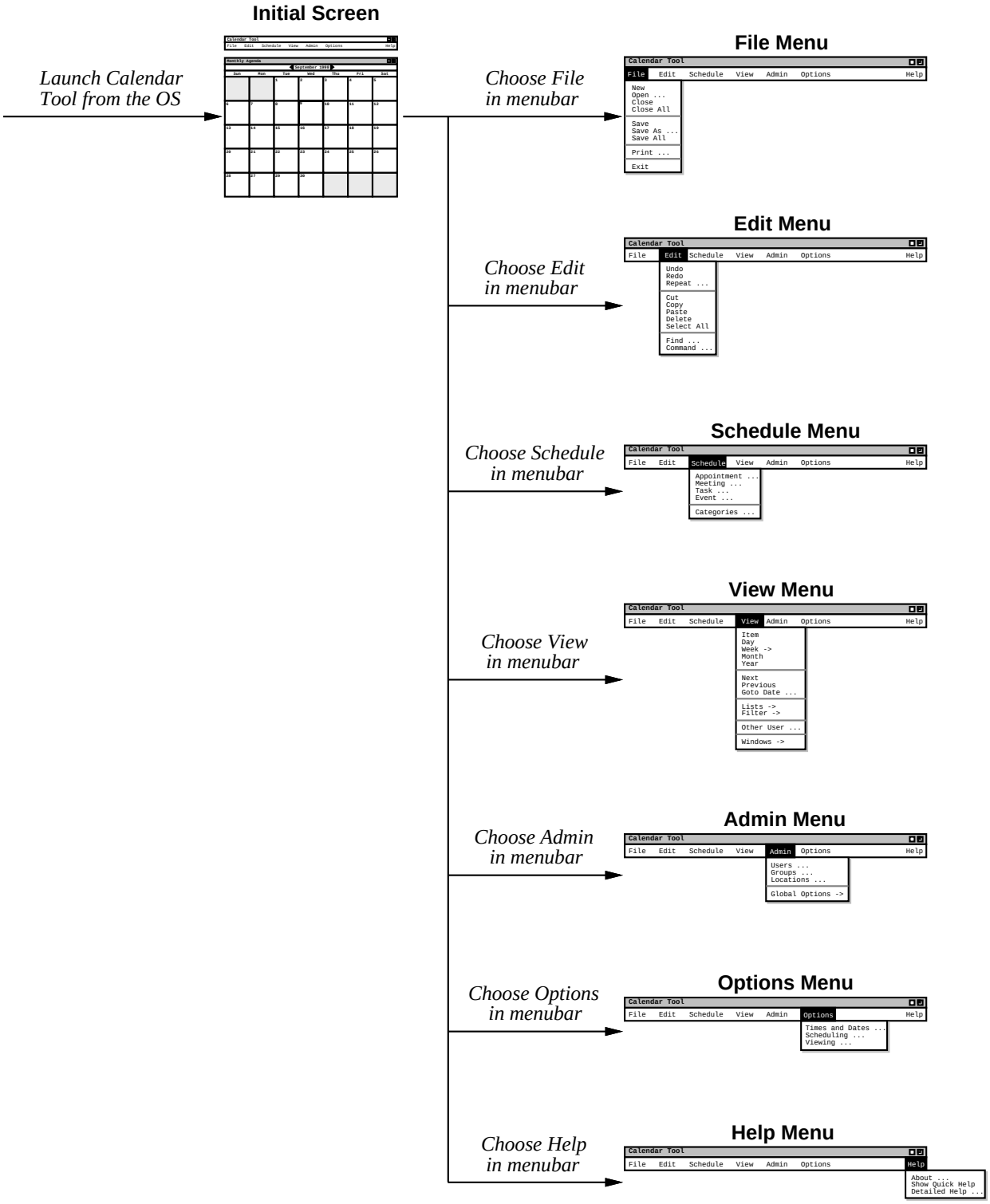
A. A potentially helpful high-level view of a GUI is a *screen map* that shows the flow of user interaction.

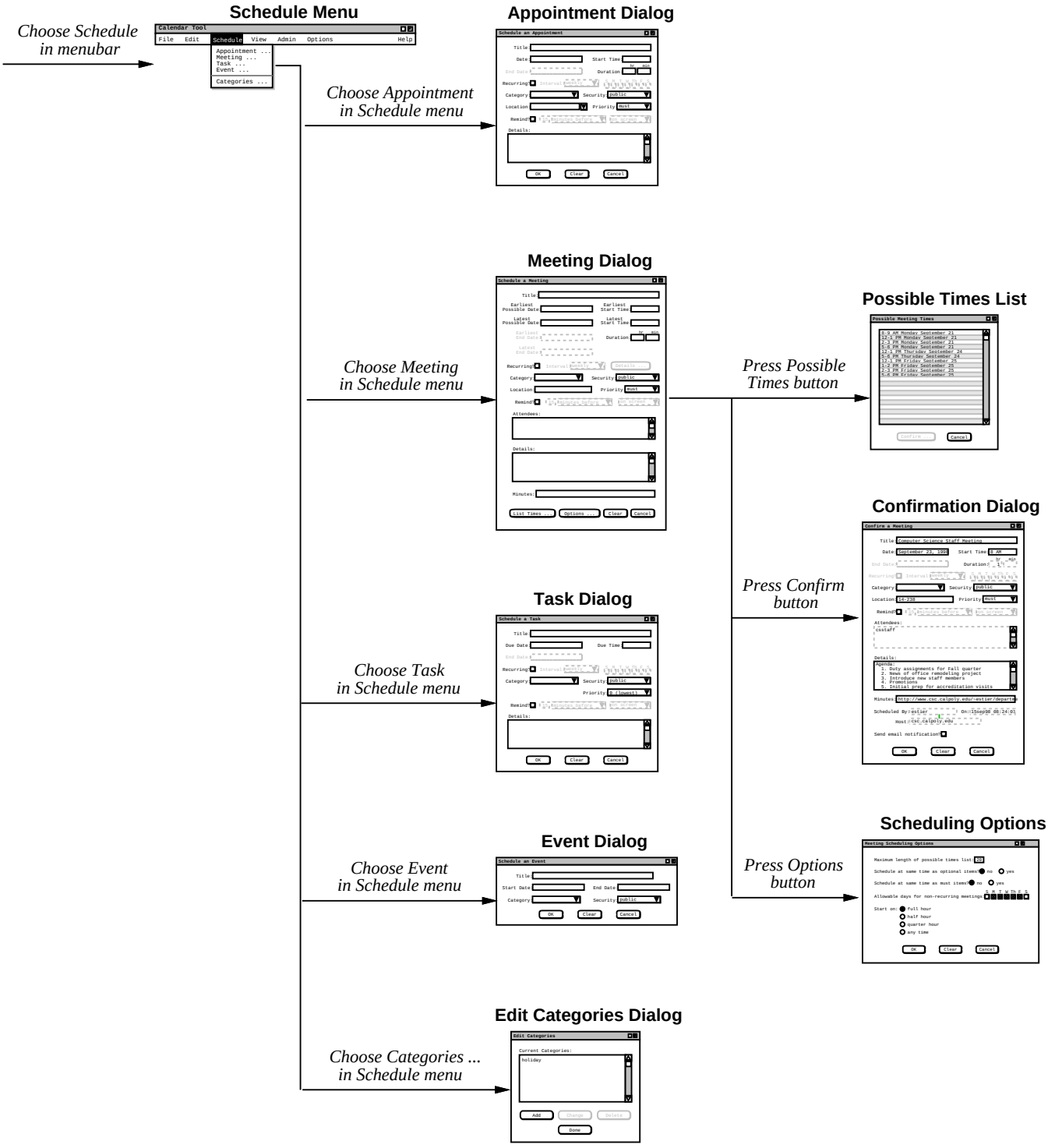
B. A screen map consists of sketches or thumbnails of interface screens, connected in a cascading tree that depicts the order in which screens appear in response to user command selection.

C. Following are a couple sample pages of a screen map for the calendar tool, showing the top-level menus and two levels below the `Schedule` menu.
   1. When sketching out screen maps, it can be useful to sketch them out initially on a large drawing area, such as a whiteboard.
   2. In the online versions, each thumbnail in the screen map can be a link to the full-sized picture in the

requirements.

D. Note that screen maps are *not required* for CSC 308, but you may find them helpful.


Initial Screen
Launch Calendar Tool from the OS
Calendar Tool
File Edit Schedule View Admin Options Help
Monthly Agenda
September 1998
Sun Mon Tue Wed Thu Fri Sat
Choose File in menubar
File Menu
Calendar Tool
File Edit Schedule View Admin Options Help
New
Open ...
Close
Close All
Save
Save As ...
Save All
Print ...
Exit
Choose Edit in menubar
Edit Menu
Undo
Redo
Repeat ...
Cut
Copy
Paste
Delete
Select All
Find ...
Command ...
Choose Schedule in menubar
Schedule Menu
Appointment ...
Meeting ...
Task ...
Event ...
Categories ...
Choose View in menubar
View Menu
Item
Day
Week ->
Month
Year
Next
Previous
Goto Date ...
Lists ->
Filter ->
Other User ...
Windows ->
Choose Admin in menubar
Admin Menu
Users ...
Groups ...
Locations ...
Global Options ->
Choose Options in menubar
Options Menu
Times and Dates ...
Scheduling ...
Viewing ...
Choose Help in menubar
Help Menu
About ...
Show Quick Help
Detailed Help ...


Schedule Menu
Choose Schedule in menubar
Calendar Tool
File Edit Schedule View Admin Options Help
Appointment ...
Meeting ...
Task ...
Event ...
Categories ...
Appointment Dialog
Choose Appointment in Schedule menu
Meeting Dialog
Choose Meeting in Schedule menu
Possible Times List
Press Possible Times button
Confirmation Dialog
Press Confirm button
Task Dialog
Choose Task in Schedule menu
Scheduling Options
Press Options button
Event Dialog
Choose Event in Schedule menu
Edit Categories Dialog
Choose Categories ... in Schedule menu

XIII. A view of requirements evolution -- Calendar Tool command menus.

A. Given below is a Subversion log report and snapshots for the evolution of the command menus of the Calendar Tool.

B. The log entries are those reported by the '`svn log`' command, which displays the revision history for a file.

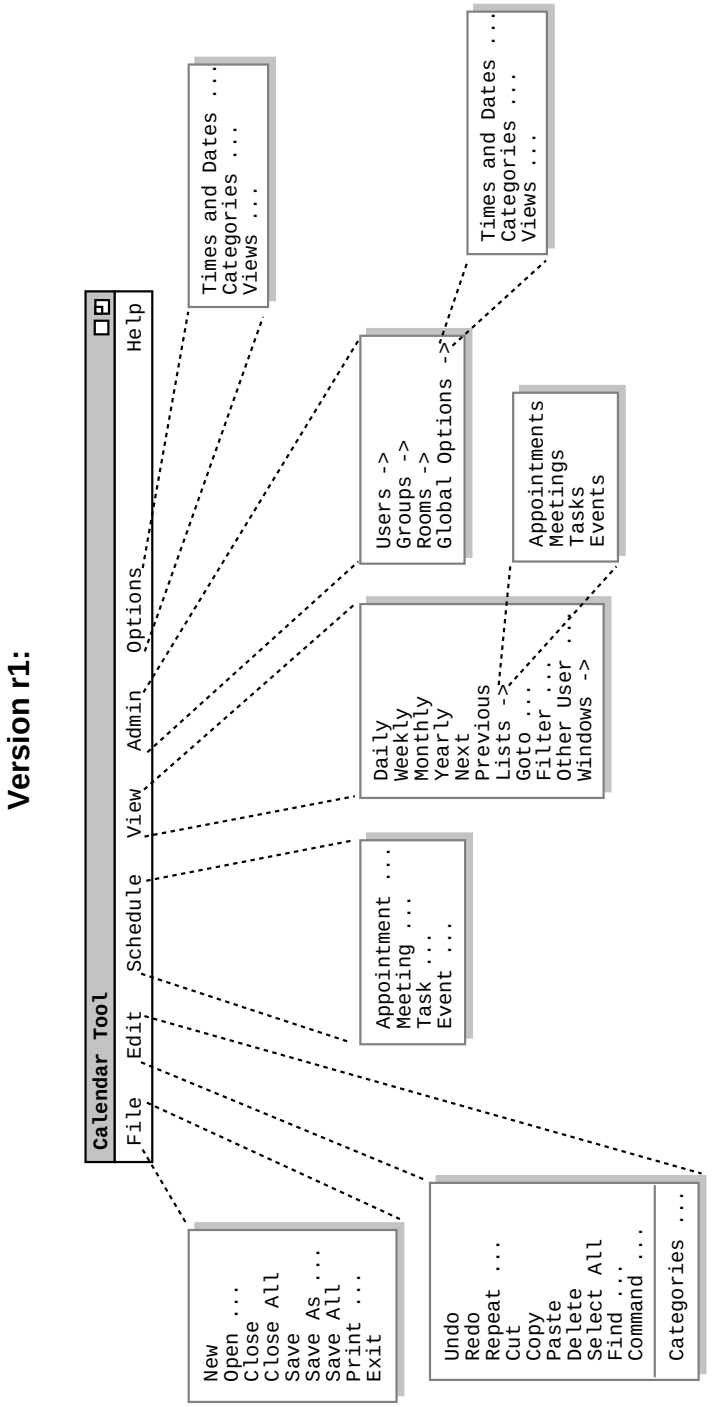
1. There is some book keeping at the top of the file that you can mostly ignore.
2. The entries describe versions 1.1 through 1.8, shown as being checked in on Friday afternoons throughout this Fall quarter.
3. The log messages are those record with the '`-m`' argument to the '`svn commit`' command.
4. The log history is shown for the file name "`menus.idr`", which is `idraw` source file for the menus screen.
5. For images, you can check in the source file and/or the generated GIF or JPEG image.

C. Here are excerpts from the Subversion log report:

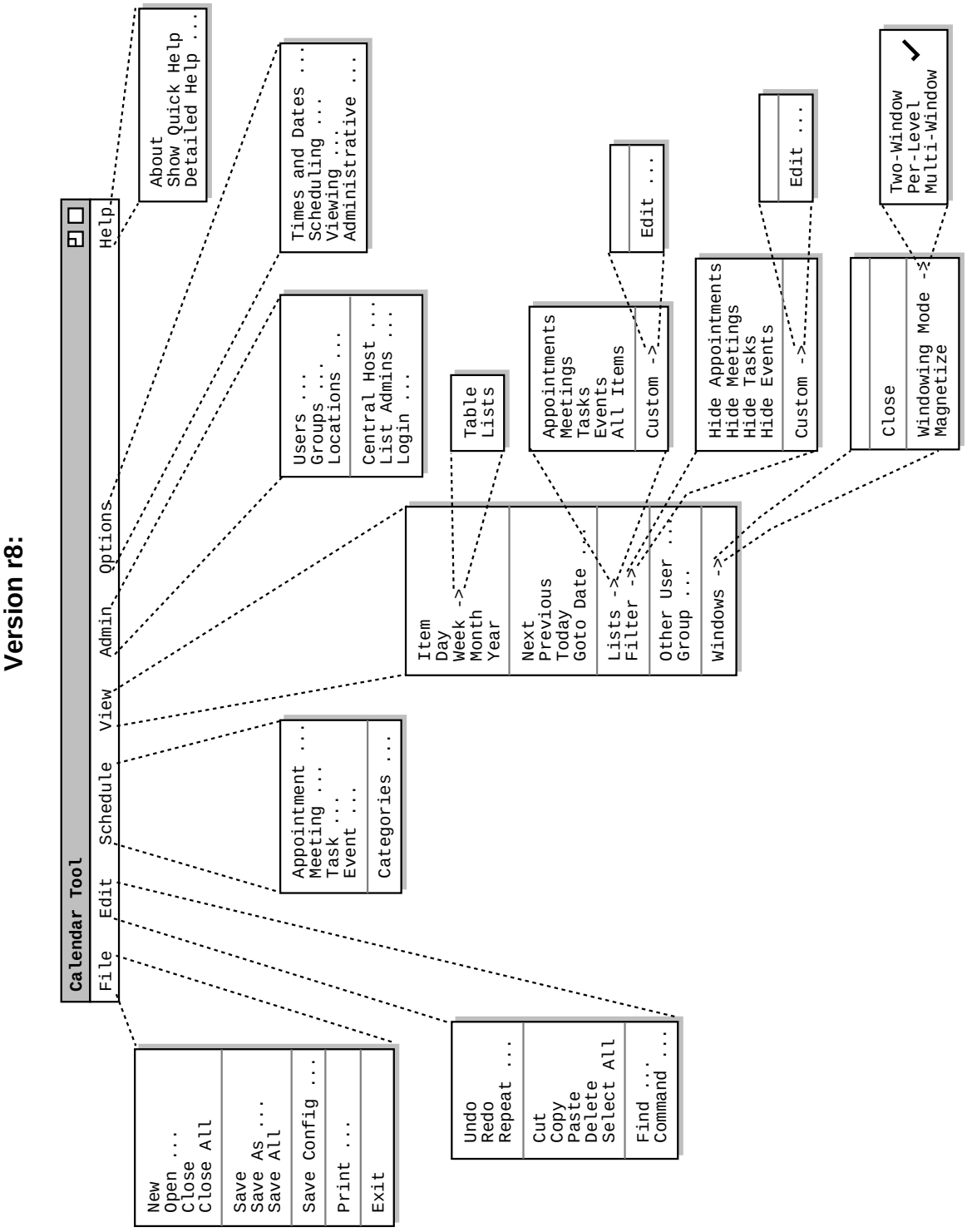
```
------------------------------------------------------------------------
r8 | gfisher | 2007-11-26 15:01:04

Replaced ‘Admin Global Options’ with ‘Central Host’, ‘List Admins’, and
‘Login’.  Also added ’View Today’, ’View Windows Close’, and ‘File Save
Config’.
------------------------------------------------------------------------
r7 | gfisher | 2007-11-19 14:20:41

Added ’Magnetize’ item to ’Windows’ menu.  Made ’Admin->Global Options’ menu
have exactly the same items as ’Options’ menu.  Added four new items to Filter
submenu.
------------------------------------------------------------------------
    ...
------------------------------------------------------------------------
r3 | gfisher | 2007-10-22 15:16:25

Added an ’All Items’ item to ’View->Lists’ submenu.  Nuked submenus for Admin
User, Group, and Room in favor of uniform dialog for each.
------------------------------------------------------------------------
r2 | gfisher | 2007-10-15 14:25:26

A nice bit of revamping to add separator lines between items, add the new Help
menu, add Sorting to View->Lists submenu, add multi-widow-mode-on-off to
View->Windows submenu, redesign Admin menu, and move entire Options menu to
Edit->Preferences.
------------------------------------------------------------------------
r1 | gfisher | 2007-10-08 14:51:35
Initial checkin.
```

D. Here are the screen shots for versions 1.1 and 1.8 in the history

Version r1:


Calendar Tool
File Edit Schedule View Admin Options Help
New
Open ...
Close
Close All
Save
Save As ...
Save All
Print ...
Exit
Undo
Redo
Repeat ...
Cut
Copy
Paste
Delete
Select All
Find ...
Command ...
Categories ...
Appointment ...
Meeting ...
Task ...
Event ...
Daily
Weekly
Monthly
Yearly
Next
Previous
Lists ->
Goto ...
Filter ...
Other User ...
Windows ->
Users ->
Groups ->
Rooms ->
Global Options ->
Appointments
Meetings
Tasks
Events
Times and Dates ...
Categories ...
Views ...
Times and Dates ...
Categories ...
Views ...

Version r8:


Calendar Tool
File
Edit
Schedule
View
Admin
Options
Help
New
Open ...
Close
Close All
Save
Save As ...
Save All
Save Config ...
Print ...
Exit
Undo
Redo
Repeat ...
Cut
Copy
Paste
Delete
Select All
Find ...
Command ...
Appointment ...
Meeting ...
Task ...
Event ...
Categories ...
Item
Day
Week ->
Month
Year
Next
Previous
Today
Goto Date ...
Lists ->
Filter ->
Other User ...
Group ...
Windows ->
Table
Lists
Appointments
Meetings
Tasks
Events
All Items
Custom ->
Edit ...
Hide Appointments
Hide Meetings
Hide Tasks
Hide Events
Custom ->
Edit ...
Close
Windowing Mode ->
Magnetize
Two-Window
Per-Level
Multi-Window
Users ...
Groups ...
Locations ...
Central Host ...
List Admins ...
Login ...
Times and Dates ...
Scheduling ...
Viewing ...
Administrative ...
About
Show Quick Help
Detailed Help ...