# B Materials

This appendix contains the materials used during the CodaConsole usability study.  In the sections that follow, you will find:

- **Procedures:** used by the experimenter to maintain consistency between participants
- **Consent Form:** required of all participants
- **Background Questionnaire:** administered before beginning the Tutorial
- **Tutorial[1]:** taught user about the interface
- **Exercises[2]:** tested user understanding of interface and ability to operate it
- **Evaluation Questionnaire:** administered after completion of the Exercises
- **Answer Key:** used by the experimenter to score the accuracy of the user's responses
- **Par Key:** used by the experimenter to score the efficiency of the user's responses

For a discussion of the goals, methods, and results of the usability study, please refer to Chapter 7, "Evaluating the Interface".

[1] This appendix contains a hard-copy version of the on-line Tutorial.
[2] This appendix contains a hard-copy version of the on-line Exercises.

# B.1 Procedures

1) Note Time: ________________
2) Reboot telos
3) Label Tapes for Tutorial and Exercises (7)
   a) Audio
   b) Video (8mm)
   c) Video (SVHS)
4) Lab Startup Procedure
5) Verify all cameras and mikes operational
6) Login as bovik
   a) kinit as mre
   b) clog
   c) xinit
   d) xterm (2 – iconify one)
   e) Tutorial -subject <ID>
7) Wait for user to arrive.
8) Note Time: ________________
9) Consent Form

   Good morning. Thank you for agreeing to participate in my user study. Please forgive me for reading from this script. Because it is important that I treat every user the same, I must read from a script.

   In today's user study, you will be using an experimental interface to the Coda File System. This study consists of four parts. First, you will begin by filling out a brief questionnaire. Then, you will go through a tutorial to teach you how to use the interface. This tutorial takes approximately 1 to 1.5 hours. After you complete the tutorial, we will take a break. Then, you will be asked to perform some exercises using the interface. These exercises are expected to take 1 - 1.5 hours. Once you have finished the exercises, you will be asked to fill out a brief evaluation questionnaire. Then, we can discuss any questions you might have.

   Before we begin, we must take care of some formalities. Please read through this consent form carefully. If you are still interested in participating, please sign it.

   **Check consent form, look for initials.**

## 10) Background Questionnaire

Thank you. Now, please read and answer each of the questions in this questionnaire. We will use this information to help us interpret the results of our study. Your responses will be kept confidential and will be reported with your participant ID number only.

**Check background questionnaire.**

## 11) Play game to demonstrate verbal protocol

While you are using our interface, we will want you to be thinking out loud. Basically, this just means that you should say what you are thinking. Both positive and negative comments are appreciated. Since many people are unfamiliar with this, we would like to demonstrate what we'd like you to do using a card game called Klondike.

Demonstrate.

Now, you try it with Othello...

Give participant feedback.

## 12) Ask user to get comfortable, but…

Let's go into the other room now. Please sit down and make yourself comfortable. If it gets too cold in here, we can move some carpeting pieces to "control" the air conditioning. If it gets too hot in here, feel free to open the door.

a) Please put on microphone(s).

b) Point out "Cheat Sheet"

You will be borrowing Harry Q. Bovik's account today. Harry uses ctwm as his window manager. You may find it useful to refer to the "Cheat Sheet" **<point to it>** to see how to manipulate windows in his environment. Before we begin, I'd like you to do four window manipulation tasks.

- Please move a window.
- Please put a window in the background.
- Please bring a window to the foreground.
- Please close a window.

c) Please do NOT start until I tell you to...

## 13) Begin Recording (3)

Ask user to start – "OK, please begin."

## 14) Note Time: __________________

## 15) Wait for user to complete tutorial.

## 16) Note Time: __________________

You have completed the first half of the study. We will now take a break of________ minutes. During this time, it is important that you not discuss this study with anyone. Let's resume at: _________

## 17) Stop Recording and Rewind (3)

## 18) Remove and File Tutorial Tapes (3)

## 19) Reset audio tape counter

20) Insert Exercises Tapes (hit record on audio) (3)

21) Start Exercises Program: Exercises -subject <ID>

22) Wait for user to arrive

I hope you had a nice break. We will now begin the exercises portion of the study.

23) Ask user to get comfortable, but…

a) Point out the xterm – "should you need to use it during the exercises".

b) Please put on microphone.

c) Please do NOT start yet.

24) Begin Recording (3)

25) Ask user to start – "OK, please begin."

26) Note Time: ________________

27) Wait for user to complete exercises

28) Note Time: ________________

29) Ask them to fill out the Evaluation Questionnaire

You have completed the exercises. We would now like you to tell us about your experience with the interface you just used. Please read and answer each of the questions in this survey.

30) Stop Recording and Rewind all tapes (3)

31) Remove, flip protect bits, and file all tapes (3)

32) Insert Debriefing tape and begin recording

33) Debriefing

Do you have any questions or comments that you would like to discuss?

34) Thank them and give them T-Shirt! \textbf{Get Initials on Contact Form}

35) Stop Recording, rewind debriefing tape, and flip protect bits.

36) Remove and File Tape

37) Check that all record tabs are either flipped or removed

38) Log out

39) Shutdown Lab

40) Note Time: ________________

41) Send mail to sesch@cs documenting lab usage time.

42) List critical observations

43) IncidentSummary <SubjectID> <IR Numbers> > ../<SubjectID>/IncidentReportSummary

44) ProcessExerciseData <SubjectID> ../<SubjectID>/ExercisesSummary.<SubjectID>

45) Cut scrolling from ActionLog and move closes to the correct locations

46) Print IncidentReportSummary, ExercisesSummary, Answers, ActionLog

47) Print Processed timing files and Raw exercise timing file.

48) Grade Answers

49) Calculate Par

50) Fill data into thesis

    a) Fill Background Questionnaire data
    b) Fill timing data for tutorial
    c) Fill timing data for exercises
    d) Fill Grades
    e) Fill Par Score
    f) Fill Evaluation Questionnaire data
    g) Duplicate all papers and file.

51) File original contact info and consent forms.

# B.2 Consent Form

## Carnegie Mellon University
## Experiment Participation Consent Form

### Purpose of Study

The School of Computer Science at Carnegie Mellon University is asking you to participate in a study of computer software. By participating in this study you will help us evaluate the software and make it easier to learn and use. The study will take place in a laboratory in Wean Hall, where you will be observed as you use the software. We will teach you how to use the software. We will ask you to use the software, while recording information about how you do so. We will ask you to fill out two questionnaires, and we may interview you. We will use the information you give us, along with the information we collect from other people, to evaluate the software and recommend ways to improve it.

The purpose of this study is to learn about the usability of our software. **Any and all problems** that you encounter during the course of the study reflect problems in the interface – not you! You represent a typical user of our system. If you encounter problems with the interface, others will as well. Please be sure to point out **any** problems, even minor ones, you experience so that we can improve the interface for future users of our system.

### Videotaping

Your work with the interface will be videotaped by one or more cameras. One or two cameras will be focusing on the screen, but will be shooting from over your shoulder. Another camera will provide a closeup of your hand on the mouse. Although no camera will be focused directly on your face, your identity could be recognized from your image on this video or from the sound of your voice as you speak (you will be wearing a microphone during the study and you will be asked to speak aloud as you use the system). You have the right to ask where the cameras are located so that you may avoid looking directly at them during the course of the study.

The researchers performing this study may want to use short clips of these videotapes (including your voice, statements, and image) for illustrative reasons during presentations of this work. These presentations include talks given to colleagues both at local seminars and thesis defenses and elsewhere at workshops, conferences, job interviews, and other meetings. The researchers will not reveal your name during these presentations -- users will only be referred to by a unique identifier (as described in the next section).

By signing this form, you give your permission for the researchers involved in this study to use the videotape(s) (including any captured image of you and any recording of your verbal comments) of your session for the purpose of evaluating this software as well as for illustrative reasons during presentations of this research as described above.

Initials: ________

(over)

## Anonymity

In order to maintain anonymity in analysis and reporting, each participant will be assigned a unique identifier, independent of name and initials. The researchers will save the data and videotape by this identifier. Only members of the research group will view the tapes in detail. Only a minimal number of people will know the mapping between identifiers and names. All presentations (both written and oral) of this work will report the results using participant identifiers.

## Breaks

We expect the study to take a total of about two hours. We have scheduled a break for you at the midpoint; however, you may take a break at any other time you wish. Merely inform the test administrator that you would like to do so.

## Compensation

Upon completion of the study, you will receive a Coda T-shirt.

## Withdrawing

You may withdraw from this study at any time for any reason.

## Questions or Concerns

If you have questions or concerns about this study, you may contact either of the following people:

| | |
|---|---|
| M. Satyanarayanan | Susan Burkett |
| Professor | Associate Provost |
| Carnegie Mellon University | Carnegie Mellon University |
| 8115 Wean Hall | 402 Warner Hall |
| (412) 268-3743 | (412) 268-8746 |

## Consent

I understand that in signing this consent form, I give M. Satyanarayanan, and his associates on this project, permission to present this work, in written, visual, and oral form, without further permission from me.

Signature: ____________________________________

Printed Name: ____________________________________

Date: ____________________________________

# B.3 Background Questionnaire

*The information in this questionnaire will help us identify users for our user study. In addition, it will help us interpret the results of the usability study. The answers you provide will be held in confidence and will be reported only as identified by your participant ID. Please circle your selection or answer each question, as appropriate.*

What is your gender?

(a) Female
(a) Male

What is your highest degree obtained?

(a) None
(b) High School Diploma
(c) Bachelors
(d) Masters
(e) PhD

What is your job title?

(a) Administrative Support Staff
(b) PostDoc
(c) Professor
(d) Research Scientist
(e) Student
(f) Technical Support Staff
(g) Other: ___________________

If you are a student,

What is your field of study? ____________________

What is your current level of study? _____________

What is your current year in that program? ______

Do you have any experience with human-computer interaction (HCI)?

(a) Yes
(b) No

If so, what?

1. During the last two weeks, how often did you use a computer?

   (a) never
   (b) once
   (c) once a week
   (d) every other day
   (e) daily

2. Which operating systems are you proficient in using? (Circle all that apply.)

   (a) DOS
   (b) Mac OS
   (c) A flavor of Unix
   (d) Other: ________________

3. a. Have you ever worked on a computer that had a windowing environment?

   (a) yes
   (b) No  Skip to question 4

   b. If so, which environments? (Circle all that apply.)

   (a) DEC Windows
   (b) MacIntosh Window Manager
   (c) MS-Windows
   (d) NeXTStep
   (e) Presentation Manager
   (f) Sun/Open Look or SunView
   (g) X Windows
   (h) Other: ________________

4. The following questions relate to your experience with Unix-based systems.

   a. Have you ever used a Unix system?

      (a) yes
      (b) no Skip to question 7

   b. During the last two weeks, how often did you use Unix systems?

      (a) never
      (b) once
      (c) once a week
      (d) every other day
      (e) daily

   c. How many years experience do you having with Unix systems?

      (a) 1 or less
      (b) 2
      (c) 3
      (d) 4
      (e) 5 or more

   d. How do you rate your ability to use Unix?

      (a) poor
      (b) fair
      (c) ok
      (d) good
      (e) excellent

5. The following questions relate to your experience using X windows and related programs.

   a. Have you ever used X windows?

      (a) yes
      (b) no Skip to question 6

   b. If so, which window manager do you prefer? _____

6. The following questions relate to your experience using various Unix applications.

a. Which editors are you proficient in using? (Circle all that apply.)

(a) vi
(b) emacs
(c) gnu-emacs
(d) epoch
(e) other: __________

b. Which one do you prefer to use? ________________

c. Which document processing tools are you proficient in using? (Circle all that apply.)

(a) scribe
(b) latex
(c) frame
(d) other: _________

d. Which one do you prefer to use? ________________

e. Have you ever maintained or compiled any Unix applications, either here at Carnegie Mellon or elsewhere?

(a) yes
(b) no

f. If so, which one(s)?

7. The following questions relate to your experience with AFS.

   a. Have you ever used AFS?

      (a) yes
      (b) no   Skip to question 8

   b. During the last two weeks, how often did you use AFS?

      (a) never
      (b) once
      (c) once a week
      (d) every other day
      (e) daily

   c. Does your primary computer have your home directory in AFS?

      (a) yes
      (b) no

   d. Do you store your e-mail in AFS?

      (a) yes
      (b) no

8. The following questions relate to your experience with Coda.

   a. Have you ever used the Coda File System?

      (a) yes
      (b) no

   b. Do you know anything about the Coda File System?

      (a) yes
      (b) no

   c. If so, what?

9. The following questions relate to your experience with the Windows 95 Briefcase.

   a. Have you ever used Briefcase?

      (a) yes
      (b) no

   b. Do you know anything about Briefcase?

      (a) yes
      (b) no

   c. If so, what?

Thank you for your time!

# B.4 Tutorial

*Order of Events:*

1. Display Initial
2. Store "BeginTime"
3. Store Begin Time for "Study"
4. Store Begin Time for "Introduction"
5. Display Introduction1
6. Display Introduction2
7. Display Introduction3
8. Display Introduction4
9. Display Introduction5
10. Display Introduction6
11. Display Introduction7
12. Display Introduction8
13. Display Introduction9
14. Execute CodaConsole Interface
15. Display Introduction10
16. Display Introduction11
17. Store End Time for "Introduction"
18. Store Begin Time for "Urgency Colors"
19. Display UrgencyColors1
20. Display UrgencyColors2
21. Store End Time for "Urgency Colors"
22. Store Begin Time for "Network"
23. Fix Space Statistics
24. Simulate weakly connected operation with respect to `scarlatti`
25. Display Network1
26. Simulate disconnected operation with respect to `scarlatti` (timed for 1.5 s. delay)
27. Display Network2
28. Display Network3
29. Fix network connection to `scarlatti`
30. Store End Time for "Network"
31. Store Begin Time for "Task"
32. Display Task1
33. Display Task2
34. Display Task3
35. Display Task4
36. Display Task5
37. Display Task6
38. Display Task7
39. Display Task8
40. Display Task9
41. Display Task10
42. Display Task11
43. Display Task12
44. Display Task13
45. Display Task14

46. Display Task15
47. Display Task16
48. Display Task17
49. Display Task18
50. Display Task19
51. Display Task20
52. Display Task21
53. Store End Time for “Task”
54. Store Begin Time for “Advice”
55. Display Advice1
56. Simulate weakly connected operation with respect to all servers.
57. Simulate a “Weakly Connected Cache Miss Advice” event with a delay of 10 seconds.
58. Display Advice2
59. Configure “Weakly Connected Cache Miss Advice” event to pop-up a window on user’s screen.
60. Display Advice3
61. Simulate a second “Weakly Connected Cache Miss Advice” event with a 13 second delay.
62. Display Advice4
63. Display Advice5
64. Store End Time for “Advice”
65. Store Begin Time for “HoardWalk”
66. Display HoardWalk1
67. Send the interface a request to begin a hoard walk.
68. Send the interface a request for hoard walk advice with a delay of 4 seconds.
69. Display HoardWalk2
70. Display HoardWalk3
71. Display HoardWalk4
72. Display HoardWalk5
73. Display HoardWalk6
74. Store End Time for “HoardWalk”
75. Store Begin Time for “Space”
76. Simulate that the RVM space is almost full (critical) with a delay of 3 seconds.
77. Display Space1
78. Fix the RVM space problem.
79. Display Space2
80. Simulate that the cache space is starting to fill (warning) with a delay of 3 seconds.
81. Display Space3
82. Fix the cache space warning with a delay of 1.5 seconds.
83. Display Space4
84. Store End Time for “Space”
85. Store Begin Time for “Tokens”
86. Display Tokens1
87. Display Tokens2
88. Display Tokens3
89. Store End Time for “Tokens”
90. Store Begin Time for “EventConfiguration”
91. Display EventConfiguration1
92. Display EventConfiguration2
93. Display EventConfiguration3
94. Display EventConfiguration4
95. Display EventConfiguration5
96. Store End Time for “EventConfiguration”
97. Simulate a “Token Expiry” event
98. Display TokenExpiry
99. Store End Time for “Study”
100. Shutdown Tutorial

| |
|---|
| Please wait to begin. |

**Screen 1: Initial**

| |
|---|
| Thank you for agreeing to participate in our usability study.<br><br>To make it easy for you to distinguish between our interface and the on-line tutorial and exercises, we have made the background color of the on-line tutorial and exercises pink. Thus, if the window is approximately the same color as the walls, it's part of the study -- not the interface you're testing.<br><br>Each screen of text is followed by an "OK" button. Click on this button to continue to the next screen (or hit <Enter>). |

**Screen 2: Introduction1**

| |
|---|
| Occasionally, the cursor changes to a picture of a clock or a picture of a mouse (a square with three rectangular boxes inside it). If this happens and it doesn't automatically go away, simply click on the X background. If it still doesn't go away, come get me. Similarly, if the program seems to hang or you believe something is seriously wrong, come get me. |

**Screen 3: Introduction2**

| |
|---|
| We will now start the tutorial. The tutorial begins by explaining what problems the Coda File System addresses and how you might use it. The tutorial then continues by introducing a new interface to Coda and explaining each portion of that interface in detail. When you need to perform an action, you will see instructions preceded by "→". Like this:<br><br>→ Please click on the "OK" button below. |

**Screen 4: Introduction3**

Coda is a distributed file system, and is a descendant of the Andrew File System that you use every day. It inherits many of the properties of AFS. When everything is working well, you would have a difficult time distinguishing between Coda and AFS.

As a user of AFS, you are probably aware that when a server goes down you cannot access data on that server. Similarly, if the network goes down, you cannot access any data in AFS. Users often find this frustrating since the machines sitting on their desks are frequently fully operational.

Coda was created to solve this problem. Our goal was to provide “high data availability”. This term simply means that we wanted to be able to continue to work even if our desktop machine was completely unable to communicate with the servers (either because the server(s) crashed or because the network connections were down). A consequence of high availability is that we can voluntarily disconnect our laptop from the network (for instance, to take it on an airplane) and still continue to access our data.

When a Coda client machine (either a desktop or a laptop) is unable to communicate with the Coda servers, it is said to be operating “disconnected”.

**Screen 5: Introduction4**

In case you weren't aware of this, both AFS and Coda cache files on your local disk after they fetch them from the AFS or Coda servers. AFS does this solely for performance reasons. Coda, as you will see, benefits in terms of performance too, but also uses this cache to provide availability.

The caveat to disconnected operation is that the user can only access data stored in the client's local cache. To achieve high availability then, we have to make sure that the cache contains the data that the user wants to access in the event of a disconnection. One might assume that a least recently used (LRU) caching policy would suffice. And, for short-term network failures, it often does. However, now that we are trying to provide high availablility for users of laptop computers, LRU simply doesn't suffice. The problem is that, while disconnected, the user may want to work on a project that they haven't been using recently.

To combat this problem, Coda provides a mechanism, called “hoarding”, that allows the user to specify and prioritize tasks that he or she plans to work on during an upcoming disconnection. For example, the user might want to hoard a task for writing a paper or hacking on a program. One aspect of our system that you will test is the interface to this mechanism.

**Screen 6: Introduction5**

Users have reported that working disconnected can be very successful. However, it does have limitations. The obvious one is that if an object is missing from the cache, the user cannot access it. In addition to this, however, any updates the user makes are not available to colleagues back at work. Furthermore, any updates the user makes are vulnerable to loss (of the laptop machine) or damage (to its disk).

Coda recently introduced "weak connectivity" to help alleviate some of these limitations. Weak connectivity allows the client to communicate with the servers over a "weak" network connection. We define a "weak" network connection as one that is characterized by low bandwidth, high latency, or intermittance. For instance, while attending a conference, you might dialin and connect to the servers over a modem to transmit updates and to service cache misses. Or, while in a meeting or lecture, you might use the recently installed WaveLAN network to operate over a radio frequency network.

**Screen 7: Introduction6**

The variations in network bandwidth seen by a client of the Coda File System can range over four orders of magnitude. The client might be using a 100 Mb/s Ethernet, a 64 Kb/s Wavelan, or a 24.4 Kb/s modem. As you might imagine, adapting to this kind of variation is difficult.

To make matters worse, some of these network connections actually cost the user money (e.g., long-distance telephone charges). It seems reasonable for the user to expect to have some control over how the network gets used. For instance, the user's preference might be to fulfill cache misses so that she can continue working. On the other hand, the user might prefer to transmit updates back to the server so that those changes are available to her colleagues. To make matters even worse, the user's preferences are unlikely to be static -- they may prefer fetching to transmitting one day and the exact opposite the next. Users need to have the ability to dynamically prioritize those activities requiring network resources.

Futhermore, Coda is a complex system. When things break, it can be difficult (especially for new users) to diagnose the problem. When preparing for a disconnected session, it can be difficult (even for experienced users) to determine what projects are available in the cache and what ones are partially or completely unavailable.

Coda users need an interface to keep them informed about the state of the system and to help them prepare for disconnection. The interface you will be using today is intended to fill this gap.

**Screen 8: Introduction7**

We have just finished giving you background on the Coda File System. We will now turn our attention to the user interface you will be testing today. The interface is called the "CodaConsole". Portions of this user study have been simulated so that we can control when certain things happen.

**Screen 9: Introduction8**

Our interface borrows from that of another complex system -- the automobile. The dashboard indicator lights of a car illuminate words or pictures to attract the driver's attention. These lights warn of problems and also indicate system state. For example, if the oil pressure is low, the "OIL" indicator illuminates. Similarly, there are indicators that inform the user that they are running low on fuel, that the car will need preventative maintenance soon, that a car door is open, that the airbags or the antilock breaks are not functioning properly, and so on. Since dashboard indicator lights have been widely adopted and since cars are so prevalent in U.S. society, we expect most of our users will be familiar with this interface.

Although the dashboard indicator lights have been widely adopted, there is plenty of room for improvement. The goal of the Coda indicator lights is to warn users before a problem is so critical that they must simply stop using Coda (whenever possible). In other words, we don't want to have an indicator light that means "pull over to the side of the road 5 minutes ago or risk serious engine damage". Furthermore, we want our indicator lights to be dynamic so that users can explore to get more information than just "something is wrong with the airbags".

**Screen 10: Introduction9**

We have just requested that the CodaConsole be started. The initialization takes a few seconds. Soon, you will see a small window appear on the right hand side of your screen a few inches below the clock. This window contains our version of indicator lights. There are nine different indicators. The names of our indicators are: Control Panel, Tokens, Space, Network, Advice, Hoard Walk, Reintegration, Repair, and Task. We have purposely made the indicator window as small as possible so that users will be willing to allocate precious screen real estate to keeping the window visible at all times. If users cover this window with other windows, they may not notice if a problem develops.

When the indicator-lights window first appears, the indicators (the text) should all be green. Green means that everything is operating normally. If something were busy or if a problem were brewing, the indicator would reflect that by changing from green to yellow. Similarly, if a critical problem were occuring, the indicator would change to red. We hope these default colors are easy for you to remember and to see. We do, however, understand that people from other cultures and people who have complete or partial color blindness may have difficulty either remembering our color scheme or seeing the actual colors. One of the first things we will do once we get started is to select colors that you can see and remember. Until then, please be patient.

**Screen 11: Introduction10**

Next, we will begin showing you how to use the interface one indicator at a time. Although some of the tutorial is likely to seem obvious to you, other parts may well be confusing. Please take the time to go through the entire tutorial thoroughly (even the easy parts). It is important that you complete the entire training session prior to beginning work on the exercises.

Note that there is a “Help” button on every screen in the CodaConsole interface (except for the indicator panel itself). The help window that appears when you click on the button provides more information about the screen on which the help button appeared.

**Screen 12: Introduction11**

First, let us find colors that you can see and remember.

→ Please double-click on the “Control Panel” indicator light. (The arrow means that you need to take some action.)

The “Control Panel” window should soon appear on your screen. You should see a list of folder tabs across the top of the window. The one labelled “Event Configuration” should be visible. Others include “Urgency Colors”, “Physical Connectivity”, “Logical Connectivity”, and “Behavior”.

→ Please click on the tab labelled “Urgency Colors”.

**Screen 13: UrgencyColors1**

You should now be looking at the “Urgency Colors” tab of the Control Panel. On this tab, you should see three sections split across the screen. The leftmost section contains a list of color names and shows you how each of them appears when written as text on a black background.

The rightmost section contains a small indicator window showing which colors are chosen for each of the three urgency levels: Normal, Warning, and Critical. By default, the color scheme is green-yellow-red.

The middle section contains three drop-down listboxes that allow you to change the color used for each of the urgency levels. The downward pointing arrow to the right of each color name signifies that a drop-down listbox is present. If you click on this arrow, a listbox will appear. The listbox will display the list of available colors, along with a scrollbar that allows you to navigate in the listbox. To select a new color, simply click on the name of the new color. Please experiment with choosing colors for each urgency level.

[For the record, I agree that this is a silly way to do this. If the widget library that I'm using allowed me to put the left list inside the drop-down listboxes, I'd do it in an instant.]

When you select a new color, you'll notice that the sample indicator panel on the right changes colors, but that the actual indicator lights do not change colors. This sample indicator panel will help you see how your choices will look next to each other. Once you are happy with your choices, you can click on the “Commit” button to have your color choices take effect. (Note that the “Commit” button is probably hidden by this screen.)

We will explore other areas of the Control Panel later in this tutorial so please don't go off and explore on your own just yet.

→ Please choose colors that are easy for you to read and remember.
→ Commit your choices.
→ Please close the “Control Panel” window.

**Screen 14: UrgencyColors2**

Now we're going to turn our attention to the “Network” indicator. You'll notice that this indicator light has changed to a yellow color. This indicates that there may be a problem. In this case, the problem is that our connection to at least one of the servers is weak (e.g., low bandwidth). Had the color been red instead of yellow, the indicator would have been showing that our connection to at least one server had been severed.

→ Please double click on the “Network” indicator for more detail regarding our connections to individual servers.

This window shows the network bandwidth to each Coda server. The meter to the right of each server's name shows the bandwidth to that server as a percentage of the maximum Ethernet bandwidth, 10 Mb/s. The color of the bar indicates whether we are strongly connected (green), weakly connected (yellow), or disconnected (red) from this server.

**Screen 15: Network1**

Soon, the bandwidth to `scarlatti` will drop to zero. Watch as the indicator light flashes red and the “Network Information” window automatically updates to show the change in `scarlatti`'s estimated bandwidth.

**Screen 16: Network2**

We have now completed our exploration of the “Network” indicator.

→ Please close the “Network Information” window.

Next, we will turn our attention to the “Task” indicator.

**Screen 17: Network3**

One of the user's primary goals in using this interface will be to prepare for disconnected (or weakly connected) operation. To prepare for disconnected operation, the user needs to specify what tasks or projects she plans to work on. Then Venus, the Coda cache manager, makes sure that the files and directories necessary for the user to work on these tasks are available through a process we call “hoarding.”

What is a task? A task is simply a set of files and directories necessary for you to work on a particular project. This set contains user data, programs, and even other tasks. For instance, in order for me to work on “writing my dissertation”, I need my thesis directory (a user data set), LaTeX (a program), and gnu-emacs (another program).

During this segment of the tutorial, you will learn how to define user data and programs, and how to combine these elements into tasks. In addition, you will learn how to specify what tasks you plan to work on.

→ Please double-click on the “Task” indicator light.

**Screen 18: Task1**

The “Task” indicator light pops up a window titled “Task Information”. This window has three main sections. The first section shows the percentage of cache space occupied. The second section lists the names of all defined tasks. The third section lists those tasks that have been hoarded by the user.

The purpose of the “Task Information” window is to allow the user to select tasks for hoarding. However, before we show you how to do that, we'd like to give you a better understanding of what a task is and how you might define one. Recall that a task is a set of files and directories needed for you to work on a particular project. It contains user data, programs, and even other tasks. Shortly, we will examine each of these components in detail.

→ Please double-click on the “writing thesis” entry of the “Tasks” list. (You may need to use the scroll bar to do this.

**Screen 19: Task2**

Double-clicking on the name of a task pops up a "Task Definition" window showing the definition of the selected task. This window has three main sections: one for subtasks, one for programs, and one for user data. Each of these three sections is split into two parts. The list on the left side shows a complete listing of all currently defined tasks, programs, or user data respectively. The list on the right side shows those definitions that are included in this task.

Let's look at the last main section---the one labelled "User Data". The list on the left shows the name of every set of user data that has been defined to date (including "advice sources", and "thesis"). The list on the right shows the subset of user data sets that have been included in the "writing thesis" task definition (just "thesis"). The other two sections, the one labelled "Subtasks" and the one labelled "Programs", are similar.

By looking at the three lists on the right, we see that the definition for "writing thesis" contains the "writing" subtask, the programs for "ScreenCapture", and the "thesis" user data.

**Screen 20: Task3**

In order to view another task definition, double-click on its name in either the "Task Information" window (as you just did) or even in the "Task Definition" window (often more convenient).

**Screen 21: Task4**

Now, let's look at the user data for the "writing thesis" task in more detail.

→ Please double-click on the "thesis" element in either the predefined user data list or the "`writing thesis' contains" list.

**Screen 22: Task5**

Double-clicking on the name of a user data set pops up a "User Data Definition" window showing the definition of the selected user data. The user data definition contains pathnames to files and/or directories. In the "thesis" definition, you should see three directories:

```
/coda/usr/bovik/thesis/
/coda/usr/bovik/thesis/dissertation/
/coda/usr/bovik/bib/
```

The first of these is the top-level of Harry Bovik's thesis area. The second is the directory in which he stores his dissertation. The third is the top-level of his bibliography directory.

You'll notice an empty entry widget after the bibliography directory. This empty entry is not part of the definition -- it just allows you to continue adding more paths to the definition.

**Screen 23: Task6**

If the pathname is a directory, the definition also contains some meta-information. The meta-information can be set to “none”, “immediate children”, or “all descendants”.

If the meta-information is set to “none”, then only the directory itself will be included. This means that you'd be able to look at the directory contents, but not at any of the children.

If the meta-information is set to “immediate children”, then the directory and its immediate children will be included. This means that you'd be able to look at the directory contents and examine any children. If the child is a file, you'd be able to read and modify its contents. If the child is a directory, you'd only be able list its contents---you would not be able to access its children.

If the meta-information is set to “all descendants”, then the directory and all its descendants will be included. (Be careful! This could be a very large amount of data!) You'd be able to look at all the children of this directory as described above, but in addition, you'd be able to access the children's children, and so on.

In this definition, all of the pathnames happen to be directories so Harry must consider what meta-information to select. In the case of the thesis area, he knows that this is a deep tree and that hoarding all of it could be a problem. Since he only needs the data relevant to writing, he has elected to hoard the `thesis` directory with the “immediate children” meta-information and the `dissertation` subdirectory with the “all descendants” meta-information option. In the case of the bibliography area, he knows that this is a shallow, small tree. He could choose either the “immediate children” option or the “all descendants” option without any problems. In this case, he has chosen “immediate children”.

**Screen 24: Task7**

Now let's explore how we can define a new user data element.

→ Please click on the black down arrow button. (It's next to the “User Data Name:” entry at the top of the screen.)

You should now see a list of existing user data definitions.

→ Please click on the one named “New....”

**Screen 25: Task8**

You should now see an empty data definition form, with the name “New...” highlighted. You'll first want to name your definition something other than “New...”.

→ Please type the name of a new set of data, perhaps “Grant Proposal” or “Writing SOSP Paper”, and then hit <Enter> to move the cursor down into the pathname definition area.

**Screen 26: Task9**

→ Please type in pathnames (feel free to make some up) that should be included in the new definition.

If the pathname you type is a file, then, when you hit <Enter>, an empty entry widget will appear, allowing you to enter the next pathname. If the pathname you type is a directory (yes, the system does check), then, when you hit either <Tab> or <Enter>, the meta-information area will become enabled, allowing you to choose the appropriate meta-information for this entry. If you used <Enter>, an empty entry widget will also appear and your cursor will be moved down. You'll need to use the mouse to select the appropriate meta-information. If you used <Tab>, you'll be able to use <Tab> and <Enter> to move around and select the meta-information respectively. If you continue to tab after selecting the meta-information, a new entry widget will appear and your cursor will also be moved down.

→ When you have completed the definition, click the "Save" button.

**Screen 27: Task10**

We have now completed our look into defining user data.

→ Please close all windows titled "User Data Definition".

**Screen 28: Task11**

Let's now turn our attention to program definitions.

→ Please double-click on the "ScreenCapture" program in the "Task Definition" window.

Double-clicking on the name of a program pops up a "Program Definition" window showing the definition of the selected program. The program definition contains one or more pathnames to executable files. In the case of the "ScreenCapture" definition, you should see four pathnames: "xpr", "xv", "xwdtopnm", and "ppmtogif". (These are the programs that we use to capture screen dumps of the windows in this interface.)

**Screen 29: Task12**

Even if a program touches other files or directories, or executes other programs, you need not include those pathnames. Once you have indicated an interest in a program, the system will track what files and directories that program requires to run, although the system will ignore any files and directories in areas of the file system that you have defined as user data. By making this distinction between program data and user data, we can let you share program definitions among different tasks. Because the system doesn't start tracking programs until you've indicated an interest in them, you'll need to run the program a few times before disconnecting from the network. Note that because this Interface is still in a mock-up form, you will be unable to do this for any program definitions you create.

**Screen 30: Task13**

A program definition may contain more than one executable. Each time you hit <Enter> after the last pathname, a new entry widget will appear allowing you to continue adding more paths to the definition. Once again, note that this empty entry is not a part of the program definition.

**Screen 31: Task14**

Let's explore how to define a new program.

→ Please click on the black down arrow button. (It's next to the "Program Name:" entry at the top of the screen.)

You should now see a list of existing program definitions.

→ Please click on the one named "New...".

You should now see an empty program definition form. You can type the name of the new program definition, "Scribe Tools", and then hit <Enter>. For the purpose of this study, assume that any program you want to include in a definition can be found in /coda/misc/bin.

→ Type in executables that should be included in the new definition
(/coda/misc/bin/scribe and /coda/misc/bin/gv).

→ When you have completed the definition, click the "Save" button.

We have now completed our look into defining programs.

→ Please close all windows titled "Program Definition".

**Screen 32: Task15**

At this point, you've had some experience with programs and user data. Now it's time to look at combining these with subtasks to define a task. Suppose your advisor has forced you to switch from LaTeX to Scribe for writing your thesis. You would probably first want to change the "writing" task to be called "writing with latex" so that you can continue to use it for other tasks. To do this:

→ Click on the black down arrow button. (It's next to the "Task Names:" entry at the top of the screen.)
→ Select the "writing" task by clicking on it.
→ Type " with latex" after the word "writing".
→ Click the "Save" button.

**Screen 33: Task16**

Your next goal would be to define the “writing with scribe” task. Now that you've saved the “writing with latex” task, you can modify it to create this new task. (Or you can go up, click the black down arrow, select the task named “New...”, and start from scratch.) We'll assume that you'll be modifying this task.

First, we should change the name of this task.

→ Click in the “Task Names:” entry area, fix the name, and hit <Enter>.

Next, we should remove LaTeX and xdvi since we can no longer use those tools.

→ Click on “latex” in the Programs area under the contains list.
→ Hit either the <Backspace> or <Delete> key.
→ Repeat for the “xdvi” entry.

Then, we will want to select the newly defined ScribeTools program definition.

→ Click on the “ScribeTools” in the Programs predefined list.

Finally, we want to save this definition.

→ Click the “Save” button.

**Screen 34: Task17**

You've just learned about user data, program, and task definitions. Now, let's go back and learn how to select tasks for hoarding.

→ Please close all the windows titled “Task Definition”.

You should now be looking at a window titled “Task Information”.
If you aren't, you should close all windows other than this one, and
then double-click on the “Task” indicator light.

**Screen 35: Task18**

Recall that this window allows us to select tasks for hoarding. The “Tasks” area contains a list of all existing tasks. To select a task for hoarding, we use a bizarre form of drag-and-drop---because the toolkit we're using doesn't support real drag-and-drop yet.

→ Click on the “writing thesis” task from this list.
→ Now, move the mouse down to the “Hoarded Tasks” list.
→ Finally, right click anywhere in this list.
→ Repeat for the “hacking advice” and “hacking venus” tasks. If you right click above/-below the name of another task, then the newly selected task will appear before/after that task.

You probably noticed that your machine beeped three times and the task indicator light turned red. The interface has just told you that a task is unavailable. This is because your newly selected tasks have not been hoarded yet. (We'll get to that shortly.)

**Screen 36: Task19**

Note that each task in the “Hoarded Tasks” list has a meter to the right of its name. This meter tells you how large this task is known to be (as measured in MB -- this number is currently mocked-up data). This meter also tells you what percentage of this task is currently available in the cache. If the meter is green, then the task is available in the cache for use during disconnected or weakly-connected operation. If the meter is red, then the task is partially or completely unavailable.

Also, notice that these two tasks have numbers to the left of their names. These numbers signify their priority during the hoard process. The item numbered 1 will have the highest priority, the one numbered 2 will have the next highest, and so forth. When you drop an item into the “Hoarded Tasks” list, the location at which you right-click in this list determines where the item is dropped and its priority. To change the priority of a hoarded task, you'll need to reselect it from the complete list of tasks and drop it in the correct position in the list of hoarded tasks. Note that each task may appear only once in the list of hoarded tasks. [If your priority numbers appear out of order, please excuse -- there's a bug that's been eluding me.]

**Screen 37: Task20**

We have now completed our explanation of how to define tasks and how to select them to be hoarded. Next, we'll discuss the “Advice” indicator and then the “Hoard Walk” indicator.

→ Please close the “Task Information” window.

**Screen 38: Task21**

To make effective use of disconnected operation, the user must define and prioritize tasks that she plans to access while away. The job of managing these tasks is sufficiently difficult that the system needs to provide as much help as possible to the user. Furthermore, the job of adapting to changes in network bandwidth of three or more orders of magnitude is sufficiently difficult that the system occasionally needs advice from the user.

Certain requests for advice require immediate attention. For example, when a user is operating weakly connected (perhaps, over a 9600 baud phone line) and she attempts to access a large file that is not cached, the time required to fetch that file can easily exceed even generous estimates of how long a user might be willing to wait. Rather than automatically performing a fetch that will take a substantial amount of time, we give the user a choice. She can fetch the file or not. Other types of advice are less critical and can be dealt with at the user's convenience. The “Advice” indicator allows the system to offer and request advice, demanding immediate attention only when absolutely necessary.

→ Please double-click on the “Advice” indicator.

**Screen 39: Advice1**

Notice that you are weakly connected to the Coda servers.

Shortly, the Advice indicator light will change to red indicating that a critical request for advice has arrived (well, been simulated). It will also show up in the "Advice Information" window at that time. This window has two main sections. The top section is a list of advice requests from the client's cache manager, called Venus. The bottom section is a list of advice offered by Venus. Each line is color-coded to indicate its urgency level. To examine a request or offer, the user simply double-clicks on its name.

→ Please double-click on the "Weak Miss" entry, read through the questionnaire, and choose "Don't Fetch".

**Screen 40: Advice2**

As you just saw, the "Weak Miss" Questionnaire shows the name of the missing object, the program that requested the object, and the estimated amount of time needed to fetch the object. The questionnaire also gives you a choice between fetching the file and not fetching it.

Since a process is waiting for this file request, it is important that this request for advice receive immediate attention from the user. One way to lessen the amount of time the system must wait for this advice (and therefore the amount of time the user must wait for this process to complete execution) is to automatically pop-up the request on the user's screen. We have just configured the system to automatically pop-up future such requests on your screen. (You'll learn how to do this a bit later.)

One complication of asking the user for advice is that she may not be available to answer the request. If that's the case, it might be better to go ahead and fetch the file than to waste valuable time waiting for an answer. Thus, the "Weak Miss" Questionnaire is written to timeout after 15 seconds if you haven't answered the question. At that time, it will ask if you need more time. The idea is that if you are simply thinking about your decision, you'll notice the second question and you can request more time. If you are unavailable, Venus can go ahead and service the demand fetch after waiting for the timeout.

→ Please close the "Advice Information" window.

**Screen 41: Advice3**

Momentarily, the same weak miss event will occur -- except this time it will automatically pop-up a window on your screen. This time, pretend you are considering your choices. In about 15 seconds, the timeout will expire and you'll see the questionnaire asking if you need more time.

→ Please wait for about 15 seconds.
→ Then select "Yes, please wait for my command".

The interface now shows the same information that was available in the original questionnaire. But it also shows that the interface (and therefore, the requesting process) is waiting for the user's decision.
→ Please make your decision---either way is fine.

**Screen 42: Advice4**

You've now explored a portion of the "Advice" indicator---the portion that allows Venus to request advice from the user under certain circumstances. It is important for the user to have the ability to influence the use of network resources when those resources become scarce. This facility gives the user that control.

The portion of the "Advice" indicator that we have not explored is the way in which Venus will offer advice to the user. We will not explore this aspect of the interface during this study.

Next, we will discuss the "Hoard Walk" indicator.

**Screen 43: Advice5**

Note that you are still weakly connected to the Coda servers.

The "Hoard Walk" indicator gives you feedback regarding the status of hoard walks, but it also gives you control over them. A "hoard walk" is the process that ensures that high priority tasks are cached in preference to lower priority tasks that have been accessed more recently. By default, the hoard walk process runs automatically every 10 minutes.

→ Please double-click on the "Hoard Walk" indicator.

You'll notice that this window tells you when the next periodic hoard walk is scheduled to begin. In addition, you can turn periodic hoard walks off, and you can request that a hoard walk begin immediately.

→ Please request that a hoard walk begin immediately.
→ Notice that the "Hoard Walk" indicator turns yellow and that the window changes to show a progress meter that indicates the status of the hoard walk.
→ Then, close the "Hoard Walk Information" window.

**Screen 44: HoardWalk1**

While weakly connected to the servers, the process that performs hoard walks may request advice before completing its duties. It requests advice when the amount of data it needs to fetch could take a substantial amount of time. The request allows you, the user, to control how much of the data gets fetched.

By default, when Venus requests such advice, the "Hoard Walk" indicator changes from yellow to red. You'll also notice the "Advice" indicator turn red as well. Since this particular form of advice is appropriate to both of these indicators, both of them indicate the request and both of them allow you to get to the advice request window. Let's examine how you can use this interface to control the amount of data fetched over a slow network connection.

→ Please double-click on the "Hoard Walk" indicator once again.

**Screen 45: HoardWalk2**

You should soon see a window labelled “Hoard Walk Advice”. In this window, you should see an area labelled “Cache Status”. This area shows you the percentage of cache files occupied (yes, there are a fixed number of cache files) as well as the percentage of cache space occupied (and a fixed number of blocks). You should also see an area labelled “Network Status”. This area shows you the average network bandwidth as a percentage of the maximum Ethernet bandwidth. Finally, you should see an area labelled “Fetch Statistics”. Ignore this area of the window for now since it is not yet fully implemented.

The main portion of the window displays a list of tasks, the cost of each task, and an indication of whether or not you've requested that this task be fetched. In addition, each task has a button indicating whether or not you've requested that the system stop asking about this task. For instance, portions of a large task may have been updated at the servers. If those portions of the task are not relevant to your current work, you may prefer that the system not update them until you connect over a strong network. Since hoard walks occur periodically, you may want the system to stop asking you about them.

For this segment of the tutorial, we will assume that you have specified some tasks (writing thesis, hacking advice, and hacking venus) that you plan to work on and that the hoard daemon is trying to make sure that those tasks are available in the file cache.

**Screen 46: HoardWalk3**

The only “task” listed initially is called “ALL Tasks Needing Data”. This entry represents all tasks that are not fully cached -- it is not an actual task. To fetch all the necessary files, click on the button under the “Fetch?” label (note the “?”). To prevent Venus from fetching these files now as well as prevent it from requesting advice about any of these files during future hoard walks (until a strong connection has been made), click on the button under the “Stop Asking?” label.

The `+' sign to the left of the task name indicates that this node can be expanded to show more detail. To expand a node, you can either double-click on the name of the node (“ALL Tasks Needing Data”), or single-click on its `+' sign. (You can compress the node in a similar fashion.)

→ Please expand the node named “ALL Tasks Needing Data”.

**Screen 47: HoardWalk4**

Here you see that three tasks need to have files fetched: “Writing Thesis”, “Hacking Advice”, and “Hacking Venus”. Notice that the costs of the three tasks do not add up to the cost of the “ALL Tasks Needing Data” task. This is because some of the tasks require the same files. You can see this fact by expanding each of the nodes. If the name of a task appears in italics, that means that this node appears in more than one location. For example, the “Editing” task appears under each of the three higher-level tasks, and the “Building CODA” task appears under both of the “hacking” tasks. The reason the costs do not add up is because the cost of fetching each of these subtasks is charged only to the first task that requests that it be fetched---the other tasks obtain it for free.

→ Go ahead and expand nodes (e.g. “writing thesis”) to see what's there–just don't select anything for fetch yet.

**Screen 48: HoardWalk5**

→ Please request that all the tasks be fetched by using the "Fetch?" button.
→ Then, click on the "Finish Hoard Walk" button in the lower right corner.

Notice that the "Hoard Walk" indicator turns yellow (the warning color) briefly while the hoard walk finishes its simulated fetching of the requested files.

This completes our overview of the "Hoard Walk" indicator.

→ Please close the window labelled "Hoard Walk Information".

**Screen 49: HoardWalk6**

Shortly, the space indicator will indicate an urgent problem.

→ Please double-click the "Space" indicator".

You should see that the RVM space is critically full (as indicated by the red color on the meter).

→ Please click on the "Help" button.

The help window will tell you what action (if any) you can take to alleviate this problem. Of course, since this event is simulated, you need not actually take any corrective action.

**Screen 50: Space1**

The RVM space problem has been solved. Notice that the indicator light reflects this new information and that the "Space Information" window disappears and then reappears displaying the new data.

**Screen 51: Space2**

Shortly, the space indicator will indicate a potential problem. Again, clicking on the "Help" button will explain the situation and tell you what action you (as the user) should take. Once again this event has been simulated and no action is actually necessary.

**Screen 52: Space3**

This demonstration of the "Space" indicator light is now complete.

→ Please close the "Space Information" window.

**Screen 53: Space4**

Let's now explore the “Tokens” indicator. For the purposes of this study, please do not click on the “Destroy Tokens” and “New Tokens” buttons. They work just as you would expect. (If you accidently destroy your tokens, you'll need to come get me.)

→ Please double-click on the word “Tokens” on the indicator panel.

**Screen 54: Tokens1**

A window titled “Authentication Information” should now be visible on your screen. This window should inform you when your tokens will expire---probably around this time tomorrow. (Coda tokens are similar to Kerberos tickets.)

**Screen 55: Tokens2**

Our brief exploration of the “Tokens” indicator light is complete.

→ Please close the “Authentication Information” window.

**Screen 56: Tokens3**

Our final segment of this tutorial shows you how to configure the way in which the CodaConsole alerts you of certain events.

What's an event? An event is simply something that happens while you are using Coda. For example, when your laptop loses connectivity to the Coda file servers, the “Operating Disconnected” event is said to have occurred. Another example of an event is the expiration of your Coda tokens.

What's a configuration? The configuration of an event tells the system what to do when a particular event occurs. For instance, do you want your machine to flash (blink) the appropriate indicator at you? Would you prefer that it NOT beep at you? Do you even want to be notified of this particular event?

→ Please double-click on the “Control Panel” indicator light.

**Screen 57: EventConfiguration1**

You should now be looking at the "Event Configuration" tab of the Control Panel. This tab has three sections. The first section (in the upper left region of the screen) contains the name of the indicator light for which we are configuring events. You can select a different indicator light by using the drop-down listbox. The second section (in the lower left region of the screen) contains the name of an event and its description. You can select a different event by using the drop-down listbox. The third section (on the right side of the screen) contains a number of configuration options described below.

The first option configures whether or not you will be notified if this event occurs. If you choose to be notified, then the remaining options can be configured to specify how you would like to be notified. For each event, you can choose how urgent you believe it to be, whether or not an information window should automatically appear on your screen, whether or not the bell should be used to alert you, and whether or not the indicator light should flash (blink) on and off to alert you.

The configuration of the "Tokens Acquired" event is shown. This event is currently configured to notify you. It has a normal urgency value (which simply means that it will use the "Normal" color you recently chose under the "Urgency Colors" tab). It does not pop-up an information window, it does not ring the bell, and it does not flash the indicator light.

**Screen 58: EventConfiguration2**

→ Under the "Advice" indicator, please select the "Read Disconnected Cache Miss" event.

Notice that this event is not configured to notify the user; therefore, the notification options are all disabled.

Please examine how the "Operating Disconnected" event (under the "Network" indicator) is configured. How would you describe this event and its configuration? The next screen gives the answer for you to verify your understanding of event configurations.

**Screen 59: EventConfiguration3**

The "Operating Disconnected" event is configured to notify the user with a critical urgency level by flashing the "Network" indicator light but not beeping or popping up a window.

**Screen 60: EventConfiguration4**

Because the other three tabs on the control panel, labelled "Physical Connectivity", "Logical Connectivity", and "Behavior", are irrelevant to this user study, we will not demonstrate their use.

→ Please close the "Control Panel" window.

**Screen 61: EventConfiguration5**

Oops! The tokens indicator shows that your tokens have expired. (Yes, this too is simulated.) This brings our tutorial to a close.

You may have noticed that we didn't discuss either the “Reintegration” or the “Repair” indicator lights. These two indicator lights have not yet been implemented. (Coming soon to an interface near you...)

The training session is now over. You can now take a 10 minute break. (Restrooms and a drinking fountain are available behind the elevator block. Vending machines are available on the 2nd floor.) When you return, we have a few exercises for you to perform using the interface. We expect these exercises to take approximately one hour.

**Screen 62: TokenExpiry**

# B.5 Exercises

*Order of Events:*

1. Start CodaConsole.
2. Display Initial.
3. Store “BeginTime”.
4. Display ExercisesIntro.
5. Begin logging user actions.
6. Store Begin Time for “Phase One”.
7. Display ChooseUrgencyColors.
8. Display IdentifyCurrentNetworkConditions.
9. Display DescribeOneOrMoreTasksUnavailable.
10. Display FixOneOrMoreTasksUnavailable.
11. Display WhenDoTokensExpire.
12. Display DescribeTaskHoardingDetails.
13. Trigger Hoard Walk Advice Request (time for the near future).
14. Display ProvideHoardWalkAdvice.
15. Save Hoard Walk Advice results.
16. Display DescribeSpaceStatus.
17. Display DescribeMissEvents.
18. Display FixMissEvents.
19. Store End Time for “Phase One”.
20. Store Begin Time for “Phase Two”.
21. Display ReadBlizzardBackground.
22. Display DefineBugTask.
23. Display DefineHeaderFilesTask.
24. Display DefineCVTask.
25. Display PrioritizeTasks.
26. Display RequestImmediateHoardWalk.
27. Store End Time for “Phase Two”.
28. Store Begin Time for “Phase Three”.
29. Display ReadIncidentBackground.
30. Display ReadGameBackground.
31. Display RunGame.
32. Create “Incident Report” Button.
33. Configure Events for Phase Three.
34. At T+100, trigger RVM to fill to a critical level.
35. At T+220, trigger RVM to return to normal.
36. At T+300, trigger a task to become unavailable.
37. At T+398, trigger a hoard walk to begin.
38. At T+403, trigger the hoard walk to complete.
39. At T+528, trigger weakly connected operation.
40. At T+643, trigger a cache miss.
41. At T+740, trigger disconnected operation.
42. At T+848, trigger a cache miss.
43. At T+942, trigger strongly connected operation.
44. At T+1031, trigger token expiry.
45. At T+1143, trigger a reintegration.
46. At T+1200, display Final.

47. Store End Time for Phase Three.
48. Store "EndTime".

| |
|---|
| Please wait to begin. |

**Screen 63: Initial**

| |
|---|
| During this portion of the user study, you will be presented with some exercises. The first set of exercises will ask you to use the interface to answer a few questions. The second set of exercises will ask you to prepare for a disconnected session. The final set will ask you to play a game while a background process accesses data in Coda.<br><br>Remember, these exercises are for us to observe a ``typical'' user of our system. If you have any problems with these exercises, it reflects poorly on our interface -- not on you! Also, please try to think aloud while you perform these activities.<br><br>Just as during the Tutorial segment of this study, the windows associated with the Exercises segment have a pink background. This should help you distinguish between the interface of the study and that of the CodaConsole.<br><br>We would like to apologize for the fact that you cannot cut and paste between the CodaConsole interface and this one. Since cut and paste is only really useful for the purposes of this study, we didn't implement it. However, cutting text from an xterm and pasting it into either interface should work just fine.<br><br>Do you have any questions before we begin? |

**Screen 64: ExercisesIntro**

| |
|---|
| Please choose colors for the urgency levels. Pick colors that you can see and remember. (If you are happy with the default colors, you need not pick new ones. |

**Screen 65: ChooseUrgencyColors**

| |
|---|
| Please describe the current network conditions. What is our connectivity to each of the servers: strongly connected, weakly connected, or disconnected? Which server currently has the best nework connectivity? Which one has the worst? |

**Screen 66: IdentifyCurrentNetworkConditions**

| |
|---|
| Under what conditions will the ``One Or More Tasks Unavailable'' event occur? Does the interface notify the user of this event? At what urgency level? Will it beep at the user? Will it pop-up a window? Will it flash the indicator? |

**Screen 67: DescribeOneOrMoreTasksUnavailable**

Since this is an important event, you want to be sure not to miss it. Please fix the settings for this event (``One Or More Tasks Unavailable'') so that it flashes and beeps, but does not pop-up.

**Screen 68: FixOneOrMoreTasksUnavailable**

At what time do your tokens expire?

**Screen 69: WhenDoTokensExpire**

What tasks are currently defined?
Which of these are hoarded?
Which of these tasks has the highest priority?
The lowest?
Which, if any, are completely available?

**Screen 70: DescribeTaskHoardingDetails**

Now, suppose you are attending a PI Workshop (attendance required to continue to receive grants from this particular funding agency) in Nowheresville, IA. You had to arrive a day and a half early to get cheap tickets and now you're bored to tears. Thankfully, you have plenty of work to do while you wait, but unfortunately, you were unable to hoard prior to leaving for the airport.

You need to hoard two tasks at this time. First, a colleague asked you to preview a submission for SOSP16. You need to pick up the latest version and read it so that you can send comments back to the authors. If you have time, you'd also like to work on a few letters of recommendation. These two tasks have already been defined (``SOSP16 Paper'' and ``Letters of Recommendation'').

You have selected two tasks to be hoarded and have recently made a connection via an (expensive) long-distance telephone line. Just a minute ago, you requested that a hoard walk be performed. Now, you notice that the hoard daemon has requested advice. Please examine the request, decide what objects should be fetched at this time, and describe your reasoning below.

**Screen 71: ProvideHoardWalkAdvice**

Please describe the current space status.
How much disk space is dedicated to the Coda cache?
What percentage of the RVM space is utilized?

**Screen 72: DescribeSpaceStatus**

Please describe the ``Weakly Connected Cache Miss Advice'' and the ``Disconnected Cache Miss Advice'' events and their configurations in the space below.

**Screen 73: DescribeMissEvents**

Please fix the settings for these events (``Weakly Connected Cache Miss Advice'' and ``Disconnected Cache Miss Advice'') to be consistent. I'd like them to be indicated at the ``critical'' urgency level such that they pop-up and flash the indicator, but do not beep.

**Screen 74: FixMissEvents**

You have now completed the first set of exercises. In the second set of exercises, you will be asked to prepare to operate disconnected for a period of a few days.

A blizzard is presently hitting Ohio. It has a history of dumping large amounts of snow (more than 3 feet in parts of Ohio so far). The National Weather Service in Pittsburgh expects the storm to hit southwestern Pennsylvania late tonight or early tomorrow morning. Roads are expected to be treacherous, and possibly closed to all but emergency vehicles.

You have lots of work to do prior to commencing your job search, so you plan to take your laptop home and work from there disconnected. You will now define a number of tasks that you would like to work on during the blizzard. (Don't forget -- you can reuse data, program, and task definitions!) Once you have defined these tasks, we will ask you to hoard them and choose appropriate priorities. Please do NOT prioritize them before that time.

As you define these tasks, we ask you to keep three things in mind. First, pretend that the machine you are running on is an i386_mach box. Second, you may assume that all executables that can be found in /coda/misc/bin (though any predefined programs listing a different path do not need to be changed). Third, because this is a mock-up, you will not be able to test your hoard profiles or run the programs. These notes are available on the card taped to the desk for your reference.

**Screen 75: ReadBlizzardBackground**

You are behind on addressing recent bugs reported by users. You'd like to take this occasion to catch up. You'll need the Coda sources, the bug reports, and the build environment.

The sources to the code that you must modify are located in three places:
/coda/usr/bovik/src/venus/
/coda/usr/bovik/src/advice/
/coda/usr/bovik/src/console/

You've squirreled the original reports away in
/coda/usr/bovik/src/bugs/

The programs you need to build Coda have been defined as ``building coda''.
The last time you hacked disconnected, you realized that you had forgotten to include gdb in your building Coda definition. It can be found in
/coda/misc/gnu-comp/i386_mach/omega/bin/gdb

Please create a task for fixing Coda bugs.

**Screen 76: DefineBugTask**

Oh, and the last time you hacked from home, you wanted to take a look at some header files that you had forgotten to hoard. They aren't really important, but it'd be nice to have them around. Unfortunately, even though each individual header file is relatively small, together they take up quite a lot of space -- so you don't want to just add them to your ``building coda'' programs since then they might be hoarded in preference to really important source files. They are located in several places:

```
/coda/project/coda/alpha/include/
/coda/project/coda/alpha/i386\_mach/include-special/
/coda/misc/c++/i386\_mach/alpha/include/
/coda/misc/tcl/common/beta/include/
```

**Screen 77: DefineHeaderFilesTask**

You need to write your CV before Tuesday when you will be talking to a representative from AT\&T Labs (assuming, of course, that the visitor can get here and that you can get to school). Since you haven't yet written your CV, you've borrowed the file for your advisor's CV to use as an example. Unfortunately, he still uses Scribe! Since you're time-pressured, you've decided to stick with Scribe. Please create a task for writing your CV. The necessary files are located in:

```
/coda/usr/bovik/personal/cv/
```

Unlike LaTex, Scribe outputs PostScript directly. To run Scribe, type ``scribe cv.mss''; it will produce a ``cv.ps'' file that you can preview with ``gv''. You may assume that the binaries for both scribe and gv are located in /coda/misc/bin.

**Screen 78: DefineCVTask**

Finally, you'll need to prioritize these tasks. Recognize that the data needed for all these tasks may or may not fit into your cache -- the priorities you choose now will decide what (if anything) gets left behind. To help you in prioritizing these tasks, we've listed all of the tasks you would like to work on during the blizzard, as well as what they are intended for. (The list below is in no particular priority order.) If you haven't already hoarded these tasks, please do so now. When you have finished, please use the space below to describe your reasons for choosing the priorities you did.

Fix bugs reported by users.
Various header files to support bug fixing.
Write your CV to give to this visitor on Tuesday.

**Screen 79: PrioritizeTasks**

Please request that a hoard walk be performed immediately.

**Screen 80: RequestImmediateHoardWalk**

During the final segment of this study, we will be simulating a more realistic usage scenario. (This final segment will take about 20 minutes.)

Please imagine that you are running your thesis through latex (or latex2e) in preparation for printing a copy of the document. This process takes a fairly long time since the document is rather lengthy and since latex requires you to run the document through twice (to update cross-references). For this reason, it is necessary for you to stick around and babysit it. Luckily, you're addicted to the computer game tetris...

As you are playing your game, you may notice problems or state changes being indicated through the indicator light interface. When this happens, you'll need to pause the game (use "p") and then click on the button labelled "Incident Report" (not yet visible) to describe the problem or state change to us on the pop-up form. Please report ALL state changes, even those changing from something bad to something good.

Note that the window containing the "Incident Report'" button will replace the window you are currently looking at until this segment of the study is completed. Please continue to play tetris throughout this segment of the study, even if it means starting a new game.

**Screen 81: ReadIncidentBackground**

Here are some excerpts taken from the tetris manual page:

The object of the game is to fit the shapes together forming complete rows, which then vanish. When the shapes fill up to the top, the game ends.

At the start of the game, a shape will appear at the top of the screen, falling one square at a time. When this shape ``touches down'' on the bottom of the field, another will appear at the top. As the shape falls, you can move it to the left or right, rotate it counterclockwise, or drop it to the bottom by pressing the appropriate keys. As you fit shapes together, completed horizontal rows vanish, and any blocks above fall down to fill in. When the blocks stack up to the top of the screen, the game is over.

The default control keys are as follows: ``j'' moves the block to the left; ``l'' moves the block to the right; ``k'' rotates the block counterclockwise; `` '' causes the block to drop; ``p'' causes the game to pause; and ``q'' quits the game. (These key bindings appear on the game screen to remind you.)

If you have any questions about the game, please feel free to ask.

**Screen 82: ReadGameBackground**

Let the game begin!

→ Please execute tetris in the xterm.
→ Please click the ``ok'' button below and then begin playing tetris.

**Screen 83: RunGame**

# B.6 Evaluation Questionnaire

*This questionnaire is designed to tell us about your experience with interface you used today. Please circle the number that most clearly expresses how you feel about each statement. Write in any comments you have below each question. If you need more room, please ask for a blank sheet of paper.*

1. The following questions are related to the tutorial.

The tutorial coverage was:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| very incomplete | incomplete | neither complete nor incomplete | complete | very complete |

Comments: ______________________________________________

b. Understanding the tutorial was:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| very difficult | difficult | neither easy nor difficult | easy | very easy |

Comments: ______________________________________________

c. Did the tutorial allow you to grasp the scope of the capabilities of the interface?

yes
no

Comments: ______________________________________________

d. Was the tutorial presented at an appropriate level of detail? (Circle all that apply.)

not appropriate, too little detail
appropriate amount of detail
not appropriate, too much detail

e. If you answered (i) or (iii) above, please provide one or more examples of where the level of detail was inappropriate.

2. The following questions relate to the indicator light interface.

a. Using this interface was:

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| very difficult | difficult | neither easy nor difficult | easy | very easy |

Comments: ______________________________________________

b. Finding the features I wanted in the indicator lights was:

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| very difficult | difficult | neither easy nor difficult | easy | very easy |

Comments: ______________________________________________

c. If you were a Coda user, how likely or unlikely would you be to keep the indicator lights visible on your screen all (or almost all) the time?

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| very unlikely | unlikely | neither likely nor unlikely | likely | very likely |

Comments: ______________________________________________

d. Did the interface present information at an appropriate level of detail? (Circle all that apply.)

not appropriate, too little detail
appropriate amount of detail
not appropriate, too much detail

e. If you answered (i) or (iii) above, please provide one or more examples of where the level of detail was inappropriate.

3. The following questions relate to hoarding, both in general and specifically in terms of your preparation for operating disconnected for the duration of the *blizzard*.

   a. Understanding the concept of hoarding was:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| very difficult | difficult | neither easy nor difficult | easy | very easy |

   Comments: ______________________________________________

   b. Preparing for the *blizzard* disconnection was:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| very difficult | difficult | neither easy nor difficult | easy | very easy |

   Comments: ______________________________________________

   c. My confidence in the task definitions I created in preparation for the *blizzard* is:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| very low | low | neither high nor low | high | very high |

   Comments: ______________________________________________

4. As we mentioned in the tutorial, our interface is adapted from the dashboard indicator lights of a car. The following questions are related to your experiences with these indicator lights.

a. Do you prefer to drive a car that uses dashboard indicator lights to one that does not?

yes, I prefer a car *with* dashboard indicators lights.

no, I prefer a car *without* dashboard indicator lights.

Comments: ______________________________________________

b. Dashboard indicator lights are:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| very ineffective | ineffective | neither effective nor ineffective | effective | very effective |

Comments: ______________________________________________

c. Please describe an instance of when you used the dashboard indicator lights in the last month.

d. Please describe a particularly memorable event when you found the dashboard indicator lights to be either particularly effective or particularly ineffective.

5. a. If you are not currently a Coda user, would you be interested in becoming a Coda user?

yes
no
Comments: ______________________________________________

b. If you are currently (or have been) a Coda user, would you be interested in using this interface?

yes
no
Comments: ______________________________________________

6. a. Please describe the aspect of the interface that you liked most.

b. Please describe the aspect of the interface that you liked least.

Thank you for your time!

# B.7 Answer Key to Exercises

This key defines the correct answer to each of the exercises in our usability study. Each bullet scores one *point*.

1. [3 pts] Please describe the current network conditions.
   - connectivity? strongly connected to all
   - best? haydn
   - worst? grieg
2. [6 pts] Describe the "One or more tasks unavailable" event and its configuration.
   - Triggered when one or more hoarded tasks become unavailable.
   - Notify – YES
   - Urgency level – CRITICAL
   - Pop-up – NO
   - Beep – NO
   - Flash – NO
3. [2 pts] Fix the settings for this event so it does not pop-up, but it does beep and flash.
   - Toggle beep.
   - Toggle flash

   ➔ Event:OneOrMoreTasksUnavailable:C:1:0:1:1
4. [1 pt] At what time do your tokens expire?
   - Tomorrow at about …
5. [12 pts] Please describe the hoarding details.
   a) What tasks are currently defined?
      - darpa grant
      - editing
      - recommendation letters
      - sosp16
      - writing

   b) Which of these are hoarded?
      - sosp16
      - recommendation letters
      - darpa grant
      - editing

   c) Which task has the highest priority?
      - sosp16

   d) What task has the lowest priority?
      - editing

   e) Which, if any, are completely available?
      - editing

6. [2 pts] Please provide hoard walk advice and describe your reasoning.
   - Only need “papers—sosp16” and “letters—recommendation”, not editing
   - Don’t need “darpa grant”
7. [3 pts] Please describe the current space status.
   - Status? All space areas are within normal parameters
   - Space dedicated to Coda cache? 97 MB
   - Percentage of RVM utilized? 33%
8. [8 pts] Describe the “Weakly Connected Cache Miss” and the “Disconnected Cache Miss” events and their configurations.
   a) Weakly Connected Cache Miss
      - Triggered by a cache miss while operating weakly connected
      - Notifies the user
      - Critical urgency level
      - pop-up – NO
      - flash – YES
      - beep – YES
   b) Disconnected Cache Miss
      - Triggered by a cache miss while operating disconnected
      - Does not notify the user
9. [4 pts] Fix the settings for these events to be consistent. I’d like them to be indicated at the “critical” urgency level such that they pop-up and flash the indicator, but that they do not beep.
   a) Weak Miss:
      - Toggle pop-up.
      - Toggle beep.

        ➔ WeaklyConnectedCacheMiss:C:1:1:0:1
   b) Disco Miss:
      - Toggle notify.
      - Toggle flash

        ➔ DisconnectedCacheMiss:C:1:1:0:1
10. [9 pts] Define a task for fixing bug reports.
    a) Data:
       - Coda sources [3 areas]
         - /coda/usr/bovik/src/venus/ with Meta: c or d
         - /coda/usr/bovik/src/advice/ with Meta: c or d
         - /coda/usr/bovik/src/console/ with Meta: c or d
       - [Bug reports]
         - /coda/usr/bovik/src/bugs/ with Meta: c or d
    b) Programs:
       - building coda
         - Add /coda/misc/gnu-comp/i386_mach/omega/bin/gdb
       - compress-related
         - [compress, uncompress, gzip, gunzip]
       - [tcl testing]
         - [wish]

c) Subtasks:
   - editing

11. [5 pts] Define a task for miscellaneous header files.
    a) Data
       - Header Files
         - /coda/project/coda/alpha/include/ with Meta: c or d
         - /coda/project/coda/alpha/i386_mach/include-special/ with Meta: c or d
         - /coda/misc/tcl/common/beta/include/ with Meta: c or d
         - /coda/misc/c++/i386_mach/alpha/include/ with Meta: c or d
12. [6 pts] Define a task for your CV
    a) Data:
       - CV
         - /coda/usr/bovik/personal/cv/ with Meta: c or d
    b) Programs:
       - scribe tools or scribe program
         - scribe
       - [gv program]
         - gv
    c) Subtasks
       - editing
13. [2 pts] Prioritize these tasks and describe your reasoning.
    - [editing can be anywhere (even first or last) or nowhere]
    - First task must be consistent with user's reasoning – otherwise CV.
    - header files must be last
14. [1 pt] Request that a hoard walk be performed immediately.
15. [3 pts] Report "Filling RVM" event.
    - Urgency Level: Critical
    - Problem: The RVM space is almost full.
    - Solution: Contact my system administrator.
16. [2 pts] Report "Fixing RVM" event.
    - Urgency Level: Normal
    - Problem: The RVM space problem has fixed itself.
    - [Solution: NA]
17. [3 pts] Task Unavailable
    - Urgency Level: Critical
    - Problem: A task has become unavailable.
    - Solution: Perform a hoard walk.
18. [2 pts] Hoard Walk Begin event.
    - Urgency Level: Warning
    - Problem: A hoard walk has begun.
    - Solution: Wait until it completes or NA
19. [2 pts] Hoard Walk End & Task Available event
    - Urgency Level: Normal
    - Problem: Nothing, a hoard walk completed; task available again.
    - Solution: NA

20. [2 pts] Report "Operating Weakly Connected" event.
    - Urgency Level: Warning
    - Problem: We are now operating weakly-connected with all servers.
    - Solution: NA (or connect to a strong network if possible)
21. [2 pts] Report "Weak Miss" event.
    - Urgency Level: Critical
    - Problem:
        - We have experienced a cache miss while weakly connected
        - [Filename: intro.tex]
        - [Requesting program: latex2e]
        - [Expected Fetch Time: 23+ minutes]
        - *Note: Since answer does not require filename, user does not lose a point if they specify the wrong one. Same goes for program and time.*
    - Solution:
        - [Decide to fetch or not; find a better network.]
22. [2 pts] Report "Operating Disconnected" event.
    - Urgency Level: Critical
    - Problem: We are now operating disconnected from the Coda servers.
    - Solution: NA (or connect over a weak or strong network if possible).
23. [2 pts] Report "Disconnected Miss" event.
    - Urgency Level: Critical
    - Problem:
        - We have experienced a disconnected cache miss.
        - [Filename: intro.tex]
        - [Requesting Program: latex2e]
    - Solution:
        - [Answer the questionnaire and reconnect to the servers ASAP.]
24. [2 pts] Report "Operating Strongly Connected" event.
    - Urgency Level: Normal
    - Problem: Nothing, we have reestablished a strong network connection to the Coda servers.
    - Solution: NA
25. [3 pts] Report "Token Expiry" event.
    - Urgency Level: Warning
    - Problem: My authentication tokens have expired.
    - Solution: Reauthenticate
26. [3 pts] Report "Reintegration Pending Tokens" event.
    - Urgency Level: Critical
    - Problem: A subtree needs to be reintegrated and my tokens are invalid.
    - Solution: Reauthenticate

# B.8 Par to Usability Study Exercises

Each of the exercises in our usability study has been given a par score. *Par* is defined as the number of actions necessary for an expert user with foreknowledge of the questions to complete the exercise. Actions are preceded by an arrow symbol. Output lines produced by the interface and taken from the action log are preceded by a bullet symbol. Actions in square brackets represent optional actions that are not counted against the user. Consecutive scrolling events are compressed into a single “scroll” action.

1) [par 0] ChooseColors.
   ⇒ [Select Control Panel Indicator Light.]
   - `Control Panel Indicator: double-click`
   - `Control Panel: Click on Events tab`
   - `Control Panel Events Tab: Selecting Tokens indicator`
   - `Control Panel Events Tab: Selecting Tokens Acquired event`
   - `Control Panel Events Tab: Selecting Tokens indicator`
   - `Control Panel Events Tab: Selecting Tokens Acquired event`

   ⇒ [Select “Urgency Colors” Tab.]
   - `NoteBookWindow:Active:urgency`
   - `Control Panel: Click on Urgency tab`
   - `Control Panel Urgency Colors Tab:  Changing Normal to ForestGreen`
   - `Control Panel Urgency Colors Tab:  Changing Warning to Gold`
   - `Control Panel Urgency Colors Tab:  Changing Critical to OrangeRed1`

   ⇒ [Choose color for “Normal” urgency.]
   - `Tix Combo Arrow: click`
   - `Control Panel Urgency Colors Tab:  Changing Normal to DarkGreen`

   ⇒ [Choose color for “Warning” urgency.]
   - `Tix Combo Arrow: click`
   - `Control Panel Urgency Colors Tab:  Changing Warning to Yellow`

   ⇒ [Choose color for “Critical” urgency.]
   - `Tix Combo Arrow: click`
   - `Control Panel Urgency Colors Tab:  Changing Critical to DarkOrange1`

   ⇒ [Commit color choices.]
   - `Control Panel Urgency Tab:  Click on Commit Button`

   ⇒ [Close Control Panel Window.]
   - `Control Panel Window: close`
   - `Control Panel Urgency Tab:  Click on Commit Button`
   - `Control Panel Physical Tab:  Click on Commit Button`
   - `Control Panel Logical Tab:  Click on Commit Button`
   - `Control Panel Behavior Tab:  Click on Commit Button`

2) [par 2] Describe Network Conditions.
   ⇒ Select Network Indicator.
   - `Network Indicator: double-click`

   ⇒ Close Network Information Window.
   - `Network Window: close`

3) [par 3] Describe "One Or More Tasks Unavailable" event and its configuration.
   ⇒ Select Control Panel Indicator.
   - `Control Panel Indicator: double-click`
   - `Control Panel: Click on Events tab`
   - `Control Panel Events Tab: Selecting Tokens indicator`
   - `Control Panel Events Tab: Selecting Tokens Acquired event`
   - `Control Panel Events Tab: Selecting Tokens indicator`
   - `Control Panel Events Tab: Selecting Tokens Acquired event`

   ⇒ Select Task indicator from combobox.
   - `Tix Combo Arrow: click`
   - `Control Panel Events Tab: Selecting Task indicator`
   - `Control Panel Events Tab: Selecting All Tasks Available event`

   ⇒ Select "OneOrMoreTasksUnavailable" from combobox.
   - `Tix Combo Arrow: click`
   - `Control Panel Events Tab: Selecting One Or More Tasks Unavailable event`

4) [par 4] Fix this event.
   ⇒ Toggle Beep.
   - `Control Panel Events Tab: Setting Notification for Notify:Beep to Yes`

   ⇒ Toggle Flash.
   - `Control Panel Events Tab: Setting Notification for Notify:Flash to Yes`

   ⇒ Commit Changes.
   - `Control Panel Events Tab:  Click on Commit Button`

   ⇒ Close Control Panel Window.
   - `Control Panel Window: close`
   - `Control Panel Urgency Tab:  Click on Commit Button`
   - `Control Panel Physical Tab:  Click on Commit Button`
   - `Control Panel Logical Tab:  Click on Commit Button`
   - `Control Panel Behavior Tab:  Click on Commit Button`

5) [par 2] Identify when tokens expire.
   ⇒ Select Tokens Indicator Light.
   - `Tokens Indicator: double-click`

   ⇒ Close "Authentication Information" Window.
   - `Tokens Window: close`

6) [par 4] Describe task hoarding details.
   ⇒ Select Task Indicator Light.
   - `Task Indicator: double-click`

   ⇒ Scroll on task list.
   - `[TaskInformationWindow:scroll:.task.panes.allTasks.f.border.frame.tasklist]*`

⇒ Scroll on selected task list.
- `[TaskInformationWindow:scroll:.task.panes.selectedTasks.f.border.frame.tasklist]*`

⇒ Close "Task Information" window.
- `Task Window: close`

7) [par 8] Provide Hoard Walk Advice.

⇒ Select Hoard Walk Indicator Light.
- `Hoard Walk Indicator: double-click`

⇒ Expand "ALL Tasks Needing Data" node.
- `HList: Select`
- `Hoard Walk Advice Window: expand node`

⇒ Expand "sosp16" node.
- `HList: Select`
- `Hoard Walk Advice Window: expand node`

⇒ Expand "recommendation letters" node.
- `HList: Select`
- `HList: Select`
- `HList: Select`
- `Hoard Walk Advice Window: expand node`

⇒ Select "papers – sosp16" to be fetched.
- `HoardWalkPendingAdviceWindow:fetch`

⇒ Select "letters – recommendation" to be fetched.
- `HoardWalkPendingAdviceWindow:fetch`

⇒ Trigger hoard walk to continue.
- `Hoard Walk Window: fetch`

⇒ Close "Hoard Walk Advice" window.
- `Hoard Walk Window: close`

8) [par 2] Describe Space Status.

⇒ Select Space Indicator Light.
- `Space Indicator: double-click`

⇒ Close "Space Information" Window.
- `Space Window: close`

9) [par 3] Describe "Weakly Connected Cache Miss Advice" and "Disconnected Cache Miss Advice" events and their configurations.

⇒ Select Control Panel Indicator Light.
- `Control Panel Indicator: double-click`
- `Control Panel: Click on Events tab`
- `Control Panel Events Tab: Selecting Tokens indicator`
- `Control Panel Events Tab: Selecting Tokens Acquired event`
- `Control Panel Events Tab: Selecting Tokens indicator`
- `Control Panel Events Tab: Selecting Tokens Acquired event`

⇒ Select "Advice" Indicator.
- `Tix Combo Arrow: click`
- `Control Panel Events Tab: Selecting Advice indicator`
- `Control Panel Events Tab: Selecting Weakly Connected Cache Miss event`

⇒ Select “Disconnected Cache Miss Advice” event.
- `Tix Combo Arrow: click`
- `Control Panel Events Tab: Selecting Disconnected Cache Miss event`

10) [par 8] Fix these configurations so that they pop-up and flash, but do not beep.

⇒ Toggle Notify.
- `Control Panel Events Tab: Setting Notification for Notify to Yes`

⇒ Toggle Flash.
- `Control Panel Events Tab: Setting Notification for Notify:Flash to Yes`

⇒ Commit Changes.
- `Control Panel Events Tab:  Click on Commit Button`

⇒ Select “Weakly Connected Cache Miss Advice” event.
- `Tix Combo Arrow: click`
- `Control Panel Events Tab: Selecting Weakly Connected Cache Miss event`

⇒ Toggle Beep.
- `Control Panel Events Tab: Setting Notification for Notify:Beep to No`

⇒ Toggle Pop-up.
- `Control Panel Events Tab: Setting Notification for Notify:Popup to Yes`

⇒ Commit Changes.
- `Control Panel Events Tab:  Click on Commit Button`

⇒ Close Control Panel Window.
- `Control Panel Window: close`
- `Control Panel Urgency Tab:  Click on Commit Button`
- `Control Panel Physical Tab:  Click on Commit Button`
- `Control Panel Logical Tab:  Click on Commit Button`
- `Control Panel Behavior Tab:  Click on Commit Button`

11) [par 36] Define a task to hoard for fixing Coda bugs.

⇒ Select Task Indicator Light.
- `Task Indicator: double-click`

⇒ Select a Task.
- `HList: Select`
- `Task Definition Window (ID=0): open`
- `Task Definition Window: Show New...`

⇒ Select “New…” Data Definition.
- `Task Definition Window: Select New... from Choices Listbox`
- `Data Definition Window (ID=0): open`
- `Data Definition Window: Show New...`

⇒ Type name and hit <Return>.
- `Data Definition Window: Show coda sources`

⇒ Type directory pathname and hit <Tab>.
- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.
- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type directory pathname and hit <Tab>.
- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.
- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type directory pathname and hits <Tab>.
- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.
- `Data Definition Window: Meta-Information Select Descendants`

⇒ Save new definition.
- `Data Definition Window: save`
- `Data Definition Window: save as`
- `Data Definition Window: save`
- `Data Definition Window: Show coda sources`

⇒ Select “New…” data definition from the combobox.
- `Tix Combo Arrow: click`
- `Data Definition Window: Show New...`

⇒ Type name and hit <Return>
- `Data Definition Window: Show coda bug reports`

⇒ Type directory pathname and hit <Tab>.
- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.
- `Data Definition Window: Meta-Information Select Descendants`

⇒ Save new definition.
- `Data Definition Window: save`
- `Data Definition Window: save as`
- `Data Definition Window: save`
- `Data Definition Window: Show coda bug reports`

⇒ Close data definition window.
- `DataDefinitionWindow(ID=0):beginclose`
- `Data Definition Window: save`
- `DataDefinitionWindow(ID=0):endclose`

⇒ Select “building coda” Program Definition.
- `Task Definition Window: Select building coda from Choices Listbox`
- `Program Definition Window (ID=0): open`
- `Program Definition Window: Show building coda`

⇒ Click in executable entry area.
- `Program Definition Element: click`

⇒ Type name of executable.
- `Program Element: <Return> begin`
- `Program Element: <Return> end`

⇒ Save program definition.
- `Program Definition Window: save`

⇒ Select “New…” program definition from the combobox.
- `Tix Combo Arrow: click`
- `Program Definition Window: Show New...`

⇒ Type name and hit <Return>.
- `Program Definition Window: Show compress related`

⇒ Type name of executable.
- `Program Element: <Return> begin`
- `Program Element: <Return> end`

⇒ Type name of executable.
- `Program Element: <Return> begin`
- `Program Element: <Return> end`

⇒ Save program definition.
- `Program Definition Window: save`
- `Program Definition Window: save as`
- `Program Definition Window: save`
- `Program Definition Window: Show compress related`

⇒ [Selects "New…" program definition from the combobox.]
- `Tix Combo Arrow: click`
- `Program Definition Window: Show New...`

⇒ [Type name and hit <Return>.]
- `Program Definition Window: Show tcl testing`

⇒ [Type name of executable.]
- `Program Element: <Return> begin`
- `Program Element: <Return> end`

⇒ [Save program definition.]
- `Program Definition Window: save`
- `Program Definition Window: save as`
- `Program Definition Window: save`
- `Program Definition Window: Show tcl testing`

⇒ [Close program definition window.]
- `ProgramDefinitionWindow(ID=0):beginclose`
- `Program Definition Window: save`
- `ProgramDefinitionWindow(0):endclose`

⇒ Click in entry area.
- `Tix Combo Entry: click`

⇒ Type name of new task and hit <Return>.
- `Task Definition Window: Show Coda bug fixing`

⇒ Select element.
- `Task Definition Window: Select coda bug reports from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select coda sources from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select building coda from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select compress related from Choices Listbox`

⇒ [Select element.]
- `Task Definition Window: Select tcl testing from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select editing from Choices Listbox`

⇒ Save task definition.
- `Task Definition Window: save`
- `Task Definition Window: save as`
- `Task Definition Window: save`
- `Task Definition Window: Show Coda bug fixing`

12) [par 16] Define a task to hoard header files.

⇒ Select a task.

- `Tix Combo Arrow: click`
- `Task Definition Window: Show New...`

⇒ Type name and hit <Return>.

- `Task Definition Window: Show header files`

⇒ Select "New…" data definition.

- `Task Definition Window: Select New... from Choices Listbox`
- `Data Definition Window (ID=1): open`
- `Data Definition Window: Show New...`

⇒ Type new name and hit <Return>

- `Data Definition Window: Show header files`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Save definition.

- `Data Definition Window: save`
- `Data Definition Window: save as`
- `Data Definition Window: save`
- `Data Definition Window: Show header files`

⇒ Close data definition window.

- `DataDefinitionWindow(ID=1):beginclose`
- `Data Definition Window: save`
- `DataDefinitionWindow(ID=1):endclose`

⇒ Select element.

- `Task Definition Window: Select header files from Choices Listbox`

⇒ Save definition.

- `Task Definition Window: save`
- `Task Definition Window: save as`
- `Task Definition Window: save`
- `Task Definition Window: Show header files`

13) [par 21] Define a task to hoard your CV.

- ⇒ Select “New…” from combobox.
  - Tix Combo Arrow: click
  - Task Definition Window: Show New...
- ⇒ Type name and hit <Return>.
  - Task Definition Window: Show cv
- ⇒ Scroll.
  - [TaskDefinitionWindow:scroll]*
- ⇒ Select element.
  - Task Definition Window: Select editing from Choices Listbox
- ⇒ Select “New…” program definition.
  - Task Definition Window: Select New... from Choices Listbox
  - Program Definition Window (ID=1): open
  - Program Definition Window: Show New...
- ⇒ Type name of program and hit <Return>.
  - Program Definition Window: Show scribe tools
- ⇒ Type name of executable and hit <Return>.
  - Program Element: <Return> begin
  - Program Element: <Return> end
- ⇒ Type name of executable and hit <Return>.
  - Program Element: <Return> begin
  - Program Element: <Return> end
- ⇒ Save program definition.
  - Program Definition Window: save
  - Program Definition Window: save as
  - Program Definition Window: save
  - Program Definition Window: Show scribe tools
- ⇒ Close program definition.
  - ProgramDefinitionWindow(ID=1):beginclose
  - Program Definition Window: save
  - ProgramDefinitionWindow(1):endclose
- ⇒ Scroll.
  - [TaskDefinitionWindow:scroll]*
- ⇒ Select element.
  - Task Definition Window: Select scribe tools from Choices Listbox
- ⇒ Select “New…” data definition.
  - Task Definition Window: Select New... from Choices Listbox
  - Data Definition Window (ID=2): open
  - Data Definition Window: Show New...
- ⇒ Type name and hit <Return>.
  - Data Definition Window: Show cv directory
- ⇒ Type pathname and hit <Return>.
  - Data Element: <Return> in pathname
- ⇒ Select meta-information.
  - Data Definition Window: Meta-Information Select Descendants

⇒ Save data definition.
- `Data Definition Window: save`
- `Data Definition Window: save as`
- `Data Definition Window: save`
- `Data Definition Window: Show cv directory`

⇒ Close data definition window.
- `DataDefinitionWindow(ID=2):beginclose`
- `Data Definition Window: save`
- `DataDefinitionWindow(ID=2):endclose`

⇒ Select element.
- `Task Definition Window: Select cv directory from Choices Listbox`

⇒ Save task definition.
- `Task Definition Window: save`
- `Task Definition Window: save as`
- `Task Definition Window: save`
- `Task Definition Window: Show cv`

⇒ Close task definition window.
- `TaskDefinitionWindow(ID=1):beginclose`
- `Task Definition Window: save`
- `TaskDefinitionWindow(ID=1):endclose`

14) [par 14] Prioritize these tasks.

⇒ Select hoarded task.
- `Task Information Window: Select selsosp16`

⇒ Unhoard task.
- `Task Information Window: Un-hoarding sosp16`

⇒ Select hoarded task.
- `Task Information Window: Select selrecommendation*-*letters`

⇒ Unhoard task.
- `Task Information Window: Un-hoarding recommendation letters`

⇒ Select hoarded task.
- `Task Information Window: Select seldarpa*-*grant`

⇒ Unhoard task.
- `Task Information Window: Un-hoarding darpa grant`

⇒ Select task.
- `HList: Select`

⇒ Hoard task.
- `Task Information Window: Hoarding Coda bug fixing`

⇒ Select task.
- `HList: Select`

⇒ Hoard task.
- `Task Information Window: Hoarding header files`

⇒ Select task.
- `HList: Select`

⇒ Hoard task.
- `Task Information Window: Hoarding cv`

⇒ Scroll to verify hoard priorities.
  - `[TaskInformationWindow:scroll:.task.panes.selectedTasks.f.border.frame.tasklist]*`

⇒ Close "Task Information" window.
  - `Task Window: close`

15) [par 3] Please request an Immediate Hoard Walk.

⇒ Select Hoard Walk Indicator Light.
  - `Hoard Walk Indicator: double-click`

⇒ Request immediate hoard walk.
  - `HoardWalkInactive:walknow`

⇒ Close "Hoard Walk Information" window.
  - `Hoard Walk Window: close`

16) [par 2] Incident: Filling RVM Space.

⇒ Select Space Indicator Light.
  - `Space Indicator: double-click`

⇒ Close "Space Information" window.
  - `Space Window: close`

17) [par 0] Incident: Fixing RVM Space.
18) [par 3] Incident: Task Unavailable.

⇒ Select Task Indicator Light.
  - `Task Indicator: double-click`

⇒ Scroll to verify just one task is unavailable.
  - `[TaskInformationWindow:scroll:.task.panes.selectedTasks.f.border.frame.tasklist]*`

⇒ Close "Task Information" window.
  - `Task Window: close`

19) [par 0] Incident: Hoard Walk Begin Event
20) [par 0] Incident: Hoard Walk Finish Walk
21) [par 2] Incident: Operating Weakly Connected.

⇒ Select Network Indicator Light.
  - `Network Indicator: double-click`

⇒ Close "Network Information" window.
  - `Network Window: close`

22) [par 1] Incident: Weak Miss Event.
  - `Advice Invokation: automatic`
  - `Advice Information Window: advice invokation`

⇒ Select answer.
  - `WeakMissQuery:select1`

23) [par 1] Incident: Operating Disconnected.
  - `Advice Invokation: automatic`

⇒ Close "Network Information" window.
  - `Network Window: close`

24) [par 2] Incident: Disconnected Cache Miss Event.
  - `Advice Invokation: automatic`
  - `Advice Information Window: advice invokation`

⇒ Select answer.
  - `[[DiscoMiss:scale]*[DiscoMiss: scale motion]*]*`

⇒ Complete questionnaire.

- `Disco Miss: done`

25) [par 0] Incident: Operating Strongly Connected.
26) [par 0] Incident: Tokens Expiry Event.
    - Do not count action against user if they type a password and click “Submit”.
27) [par 4] Incident: Reintegration Pending Event.

⇒ Select Tokens Indicator Light.

- `Tokens Indicator: double-click`

⇒ Close “Authentication Information” window.

- `Tokens Window: close`

⇒ Select Reintegration Indicator Light.

- `Reintegration Indicator: double-click`

⇒ Close “Reintegration Information” window.

- `Reintegration Window: close`