REPLACING A NETWORKING INTERFACE "FROM HELL"

Roxanne F. Bradley
Hewlett-Packard Company
19420 Homestead Road, MS 43LE
Cupertino, CA 95014
408/447-4240
roxanne_bradley@hp6600.desk.hp.com

Linn D. Johnk
Hewlett-Packard Company
19483 Pruneridge Avenue, MS 48NA
Cupertino, CA 95014
408/447-0804
ljohnk@cup.hp.com

ABSTRACT
A multidisciplinary design team at Hewlett-Packard (HP) has successfully designed a new user interface for a network troubleshooting tool. Users felt that the new interface let them focus on the task of network troubleshooting, thus freeing them from the details of the interface and its underlying implementation. The design team believes that the success achieved is due to the process used and the multidisciplinary aspect of the team.

This design review describes the process followed by the design team, the difficulties encountered, the results obtained from a comparative evaluation of the new and existing product interfaces, and the lessons learned.

KEYWORDS
User-centered design, usability release criteria, usability inspections, comparative usability testing.

INTRODUCTION
Networking products are typically difficult to learn and use. This is because, by definition, networking products span multiple systems, multiple communication methodologies, and, probably, multiple geographic areas. Representing these complex concepts and data structures makes developing interfaces for networking products quite difficult.

This project involved redesigning the existing interface for a network troubleshooting tool -- Network Tracing and Logging (NetTL). NetTL is a facility that is used by many of Hewlett-Packard Company's (HP's) networking products. It provides the user with a consistent way to gather information about networking products to solve problems with the network. A survey of both external and internal users found that using NetTL was so difficult that many users refused to use it or used only a small fraction of its capabilities. In fact, some users have referred to this interface as the interface "from hell."

By analyzing the users of NetTL and the tasks that NetTL needs to support, the design team found that:

- Users are under pressure when solving network problems.
- Problem resolution must be as quick as possible.
- Users need to use this tool only infrequently.
- Users do not have knowledge of the underlying networking products' structure (i.e., subsystems).

The team found that the current interface (a command line) did not meet the needs of the user because of its steep learning curve and high cognitive (or mental) load. To illustrate, the NetTL user had to understand:

- Two cryptic command line interfaces with multiple, non-intuitive options (there was one interface for capturing data and a completely different one for reporting data).
- The nettl file structure, because files have to be modified to get products to work as wanted.
- The underlying structure of networking products since information about products was presented in terms of subsystems with no references to the associated products.

Figure 1 shows the steps, using the current interface, to perform a fundamental task -- changing the type of information gathered about a networking product (in this case, OSI) from Disaster and Error messages to Disaster, Error, and Warning messages.

Because of the steep learning curve and heavy mental load placed upon the user when using the current interface, the user had no time to focus on the networking problem. The team agreed that a well-designed graphical user interface would decrease the learning curve and lessen the user's mental load and thus significantly improve the accuracy and speed with which the user could get networking information. Figure 2 shows the steps, using the graphical interface, to perform the same fundamental task as outlined in Figure 1.

1. The user must **remember or look up** the command "nettl -ss".
2. The user types "nettl -ss" and receives the output as shown below.
3. The user must **remember or look up** the subsystems associated with OSI. NOTE: At this point the user very often would give up and call HP. Subsystems are: ACSE_US, CM, EM, HPS, SHM, ULA_UTILS.
4. The user must **remember or look up** the command to change the logging levels:
   nettl [-l class [class...] -e subsystem [subsystem...]]
5. The user must type the command.
   nettl - l disaster error -e acse_us cm em hps shm ula_utils
6. To ensure the changes are correctly made, the user must **remember** and type "nettl -ss" and **remember** the subsystems to look for changes on the display.

```
Logging Information:
Log Filename:                                   /usr/adm/nettl.LOG0*
Max LOG file size(Kbytes):      1000            Console Logging:        Off
User's ID:                      0               Buffer Size:            8192
Messages Dropped:               0               Messages Queued:        0

Subsystem Name:                 Log Class:
FDDI                                                            DISASTER
ACSE_US                                                         DISASTER
CM                                                              DISASTER
EM                                                              DISASTER
HPS                                                             DISASTER
SHM                                                             DISASTER
ULA_UTILS                                                       DISASTER
NS_LS_BUFS                                                      DISASTER
NS_LS_CASE21                                                    DISASTER
NS_LS_COUNT                                                     DISASTER
etc.

Tracing Information:
Trace Filename:                                 /usr/adm/nettl.TRC*
Max Trace file size(Kbytes):    500
User's ID:                      0               Buffer Size:            69632
Messages Dropped:               0               Messages Queued:        0

Subsystem Name:                 Trace Mask:
HPS 0x30000000
```

Figure 1. Current NetTL Interface Steps for Changing Logging

1. The user must **remember or look up** the command "nettladm".
2. The user types "nettladm." On the resulting display (see below), the user finds the product of interest in the window, highlights all of the subsystems associated with the product of interest, and selects "Modify Logging..." from the "Actions" menu.
3. In the resulting dialog box (see below), the user selects the appropriate logging levels, and selects "OK". NOTE: The user can change the console notification here as well. With the current interface, the user would have to use a different tool *and* modify a file to change the console notification.
4. The logging screen is returned, where the user can quickly verify that this information is correct.

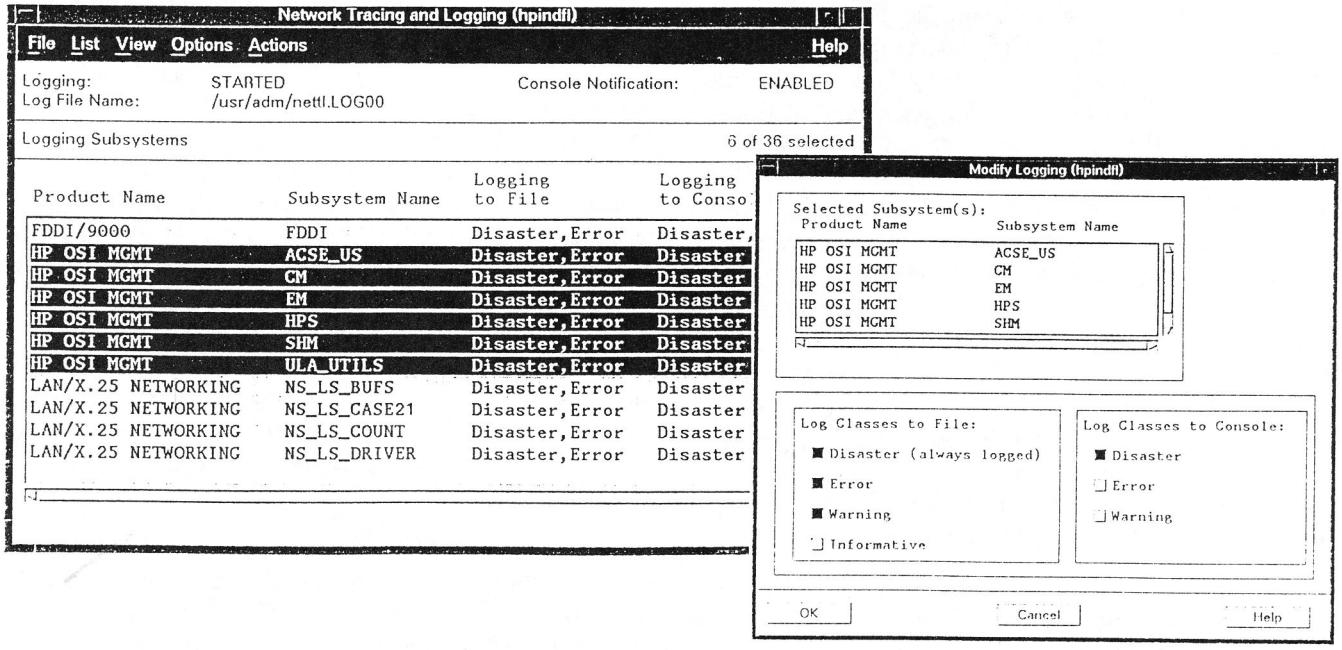

Figure 2. New NetTL Interface Steps for Changing Logging

THE DESIGN PROCESS

Creating a Team

The design team consisted of a Human Factors Engineer, a Learning Products Engineer, an Interface Development Engineer, and a Software Development Engineer. The Human Factors Engineer was responsible for leading the team in applying the user-centered design process. During this process each member of the team was responsible for different points of view:

Human Factors Engineer	Responsible for applying Motif interface design rules and HP's internal Motif design implementation, and ensuring that human factors principles were applied.
Learning Products Engineer	Responsible for ensuring that the design required minimal, if any, hardcopy documentation and minimal online help.
Interface Development Engineer	Responsible for ensuring that the design could be implemented with the tools chosen and for developing prototypes.
Software Development Engineer	Responsible for ensuring that the design could be implemented given the current NetTL technology.

Although the entire team was committed to making the new interface easy to use, only the Human Factors Engineer had any experience with the user-centered design approach. Each member of the team began the design process with a different perspective about what the process and their role in the process would be. The Software Development Engineer believed that the interface was independent from the product code and that the team did not need to be concerned about any changes being made to the code. The Interface Development Engineer already had a prototype constructed when the team was put together and at times expressed concern that spending a great deal of time on the users and their tasks was not getting the design done. The Learning Products Engineer had never been involved this early in a design before and wasn't sure what a Learning Products Engineer's role in the process was.

Agreeing on a Process

After discussing the various ways user-centered design could be done, the team initially decided that they would do three major iterations of the design. Each of these iterations included a design/redesign, a prototype, and an evaluation. The first two evaluations would be inspections, while the final evaluation would be a user performance test.

The manager of the Software and Interface Development Engineers supported the user-centered design process, but did not understand that code would be developed later in the process than anticipated. Several times throughout the process the manager had to be reminded that a user-centered design approach focused on design first and then coding.

Understanding the Product

The Software Development Engineer explained to the team that NetTL is a facility that can be used for tracing and logging by HP-developed networking products. Networking products consist of several subsystems. During operation, these subsystems may encounter some activity that is designated as an exception to the normal rules of operation. Since exceptional or abnormal events may indicate that the product is not operating as it should, the user needs to be notified of these events.

The purpose of logging is to record or log events as they occur. Events fall into four major categories: Disaster (always logged), Warning, Error, and Informative. To obtain notification of desired event types, the user must specify the types of events about which he or she wishes to be notified. The user can also specify how that notification should take place. Events are recorded in a log file; notification of events may also be sent to the console monitor if desired. Once information has been recorded in a log file, the user may retrieve all or part of it by setting up appropriate search criteria. Capturing and retrieving the right information is a critical first step in the troubleshooting process.

Tracing differs from logging in that it is not triggered by an event. Rather, tracing records the activity of a given subsystem. As with logging, the tracing user can specify what information is captured for later perusal, as well as the search criteria to apply to the captured data. Tracing is typically done once a problem has been isolated to one or more subsystems.

Analyzing Users and their Tasks

The Interface Development and Software Development Engineers understood the HP internal users of NetTL. The Human Factors Engineer understood HP's networking customers and their environments. The Human Factors Engineer began leading the team through the user and task analysis by facilitating discussions of: a) who all the users in a networking environment were, b) what the networking tasks for those users were, and c) how the NetTL-specific tasks mapped back to the networking user population. From that analysis the team determined the primary and secondary users and the primary and secondary tasks for those users.

The primary users of the logging portion of NetTL are system and/or network administrators and HP field support personnel. The secondary users of logging are network application developers and HP development and call support personnel. The experience with networking for these primary and secondary users ranges from little or

none to considerable. All these users primarily use logging to obtain notification of desired event types and to troubleshoot problems.

For the tracing portion of NetTL, the primary users are network application developers and HP development and call support personnel, while the secondary users are system and/or network administrators and HP field support personnel. The secondary users of tracing are most likely to use that portion of the facility under the guidance of a primary tracing user. Troubleshooting is the primary task that tracing supports. Tracing facilitates troubleshooting by helping the user find out as much as possible about what is happening on the network.

Since it had been determined in a survey that the primary users of logging were completely unsuccessful using NetTL, the design team decided that simplifying logging tasks for these users would be the primary goal of the design team. Thus, the design team determined the key task objectives for the first release of the new interface were:

- Ability of the primary logging user to log events of interest.
- Ability of the primary logging user to retrieve logging data necessary to perform basic troubleshooting tasks.
- Ability of the primary logging user to identify and correct problems resulting from loss of messages by the logging facility.
- Ability of the primary tracing user to guide a secondary tracing user successfully through a trace.

The team learned to use this information about the users and their tasks to make design decisions. As the team became more skilled, the role of usability advocate increasingly became a shared one, rather than the responsibility of the Human Factors Engineer alone.

Developing Release Criteria

Release criteria are used to set goals for products. These goals help the team make decisions about added functionality, interface design trade-offs, and whether the product is ready to be released. Release criteria also describe what the user experience should be like when performing tasks using a product.

Before developing the release criteria, the team evaluated what impact the knowledge of the users and their tasks could have on the design of NetTL. The team agreed that: 1) notifying users of networking problems is critical, 2) enabling users to solve networking problems in a timely manner is critical, and 3) enabling users to solve problems without having to call HP support is critical.

The team understood that, for the new NetTL interface to be successful in meeting user needs, the learning curve, the mental load while performing tasks, and user time on task would all need to be reduced. At the same time, the user's chance of success and level of satisfaction with the product would have to be increased. Based on this understanding, the team set the release criteria in Table 1.

Measure	Target	Minimum Acceptable
Time on task	≤ 2 minutes	≤ 10 minutes
Success rate	100%	80%
Percentage using hardcopy	0%	20%
Satisfaction rating	≥ 7 on 9-point scale	≥ 5 on 9-point scale

Table 1. Release Criteria

Target values represent ideal or "stretch" goals, while the minimum acceptable is the level of performance that must be achieved in order to release the product. Because these were product release criteria, this meant that, before the product could be released, at least the minimum acceptable criteria had to be met for each key task objective.

Most of these measures are well understood within HP as user needs but had never been used as product release criteria. Because each key task had to meet each of these release criteria, the criteria were considered very aggressive, if not impossible, by other HP development teams in similar business areas.

Iteration 1

The first iteration involved designing the interface, developing a paper prototype, and evaluating that prototype.

The design. For this design, the team relied upon the user and task analysis to make design decisions. Each member of the team contributed to the design in unique ways based upon their expertise. For example, the Learning Products Engineer evaluated each design idea and determined what the documentation requirements would be. If hardcopy documentation would be required to support the design idea, the Learning Products Engineer informed the team and the team modified the idea or devised a new approach.

As the team designed, a paper prototype was drawn with descriptions of the fields and navigation that would take place in the context of performing key logging and tracing tasks.

The evaluation. For the evaluation of the design, the team performed a usability inspection. A usability inspection is an evaluation process in which people review the interface using typical task scenarios. The inspectors raise concerns

about how the interface or product works, and these concerns, as well as any ideas or suggestions are logged. The design team then reviews the concerns and determines the root cause for each concern. These root causes are then examined and the interface redesigned based upon what the team believes will fix the root cause. The inspection team for the paper prototype consisted of internal users of the existing product.

Most of the inspectors' concerns with the interface were about terminology and the purpose of an action or screen. These concerns indicated a mismatch between the interface and the user's mental model of NetTL. It was obvious that there were some major problems with the design. In fact, most of the inspectors were unable to complete the task scenarios within the allotted time.

Iteration 2

The second iteration involved redesigning the interface, developing a prototype, and evaluating the design.

The redesign. The root cause of many concerns raised during the inspection was that information was not being presented in a format users could understand. Another root cause was that the design too closely imitated the existing product by arbitrarily separating tasks, such as capturing the data from retrieving it. The team agreed that the new graphical user interface should hide more of the complexities of the existing product.

For example, the configuration of logging to the log file and logging to the console was one example where the original design arbitrarily separated tasks by too closely following the existing product. In the redesign, these configuration activities were "integrated" by modifying the interface so that the two tasks could be performed on the same screen.

The Prototype. After many design meetings in which the team discussed the user's mental model and iteratively redesigned each screen, the redesign was complete. The redesign was documented and a prototype developed. Navigation was possible in the prototype, and, although data could be manipulated on the screen, it was not saved, nor did data manipulations affect other screens. With this design, the user was able to see the objects of interest upon entry into the product. Also, the actions allowed were related to the objects on the screen.

Development of the prototype is another example of where the multidisciplinary team was of value. As the Interface Development Engineer developed the prototype and found technological problems implementing the design, the Interface Development Engineer brought solutions to these problems to the team. The team discussed the proposed solutions and either agreed to them or modified them.

The evaluation. Once the prototype was complete, another usability inspection was performed. The inspectors from the first inspection participated.

With the redesigned interface, the inspectors were able to perform more of the tasks successfully. There were fewer concerns during this inspection than in the first, and the concerns focused more on the functionality than on the terminology. From an analysis of the concerns raised by the inspectors, it became clear the interface still needed to provide the user an obvious way to access and accomplish frequently performed tasks.

Iteration 3

The third iteration involved redesigning the interface and updating the prototype.

The redesign. Addressing the concerns from the second inspection did not necessitate a complete redesign as was the case with the first inspection. These concerns caused the team to rethink some of the information on the screen and how it was presented. One of the root causes for the concerns identified was the underlying operation of the product. For example, console notification was determined by setting the events for all products. If Disaster events were being reported to the console, and the user wanted to be notified of Disaster and Error events for a particular subsystem, Disaster and Error had to be turned on for all subsystems. The team agreed that console notification should be determined on a subsystem basis. This required modifications to NetTL by the Software Development Engineer.

The Software Development Engineer was willing to make the changes to console notification so that it could be turned on for individual subsystems, as well as other modifications necessary to improve usability. This need to make internal changes to improve usability and navigation underscores the fact that a user interface is not something added on to the product as an afterthought or at the end of a development cycle.

The remainder of the changes involved fine tuning the interface. This was done fairly quickly for most screens.

The Prototype. Based on the comments received, the team decided that no more major iterations were necessary, it was time to evaluate the product in terms of the release criteria. For the evaluation of the release criteria, HP customers would use the prototype. The prototype reflected the new design and simulated the manipulation of real data.

MEETING THE RELEASE CRITERIA

The design team decided that a timed usability test was the best way to determine whether the criteria had been met. The team concluded a timed test could determine how users of the graphical and command line interfaces would

perform relative to the release criteria and whether there would be a difference in user performance based on the type of interface used. To avoid the risk of introducing bias into the testing, the design team selected a Human Factors Engineer who was not part of the design team as the test administrator.

Test Design

Test Participants. Twenty system and/or network administrators were recruited from the HP customer base to participate in the test sessions. All test participants indicated they were at least sometimes responsible for monitoring network products and doing preliminary troubleshooting. These individuals were either novice or occasional users of the existing NetTL interface.

Half of the participants performed logging and tracing tasks using the new graphical user interface (GUI); the other half of the participants performed the same tasks using the existing command line interface (CLI).

Test Environment and General Procedures. The tests were conducted in the HP Cupertino Site Usability Lab. Participants were videotaped while performing representative tasks. Since these were timed tests, the think-aloud method was not employed. Users were told at the outset that if difficulties were encountered, they should attempt to work through them on their own as best they could, but not to the point of frustration. A maximum of ten minutes was allowed per task (based on the minimum acceptable criteria for time on task).

Three logging and one tracing task comprised the test session. The tasks were as follows:

1. Changing the logging level for a particular networking product from Disaster to Disaster, Warning, and Error.
2. Monitoring the log file to determine that the appropriate information was being logged for the networking product of interest.
3. Determining if there was any problem with logging information being "dropped" from the log file and, if such a problem existed, resolving it.
4. The tracing task was a scripted, simulated support call.

Following completion of the final task scenario, a posttest attitude survey was administered to measure reactions to the product. After finishing the questionnaire, participants were debriefed, both to gain an understanding of the participant's ratings as well as to obtain retrospective feedback on the experience of using the product and any suggestions for improvements.

Data Analysis. To assess performance relative to the release criteria, both quantitative and subjective measures were obtained. The following three measures of user performance were taken and analyzed for each task tested: 1) time required to complete tasks, 2) successful completion, and 3) whether hardcopy documentation was required to perform a task, where merely reaching for the manual was counted as requiring it.

Satisfaction ratings were collected and analyzed for the interface in general. The ratings were made on a 9-point scale and included the following:

- Separate averages for each of seven attributes [Easy to learn, Easy to use, Flexible, Power (i.e., adequate functionality to accomplish tasks), Stimulating, Satisfying, Wonderful].
- Composite average of the seven attribute scores, representing an overall perception of usability.

Test Results

On all measures, performance with the GUI exceeded the minimum acceptable release criteria while performance with the command line interface seldom did, specifically:

- The GUI met the minimum acceptable criteria for all tasks. It also met the target criteria for satisfaction and hardcopy usage for all tasks.
- The CLI met the minimum acceptable criteria for time and success on only one task: the simulated support call task (a task users did not have to perform entirely on their own). For the Power attribute of the user ratings, the CLI also met the minimum acceptable.

Time Results. Time on task using the GUI (4.1 minutes averaged over all test tasks) met or exceeded the minimum acceptable criteria of ≤ 10 minutes. Participants using the command line interface required more time to complete tasks. In fact, except for one task, all command line users required more than the allotted 10 minutes, so the CLI did not meet the minimum acceptable time criteria.

The following comments from representative users reflect the difference in time:

- CLI user: "After an hour [on all test tasks] I still couldn't get it to work. I felt another hour wouldn't help. It looked like it would take days."
- GUI user: "The time it took to do things was reasonable; it only took a minute or two."

Success Results. Participants using the GUI were considerably more successful than CLI users (GUI -- 78% success rate averaged over all test tasks; CLI -- 16% success rate averaged over all test tasks). The GUI success rate is not statistically different from 80% so the GUI met the minimum acceptable success criteria despite the presence of two critical defects. Those defects have been fixed, and subsequent user feedback suggests the success rates for the GUI would now be much higher. The CLI success rate

value reflects the one relatively simple task that some participants were able to complete successfully.

These user comments describe the difference in the perception of task success:

- CLI user: "Nothing was easy. I don't think I succeeded in any task."
- GUI user: "It was satisfying to use. I felt confident about getting on the road to the solution."

Hardcopy Results. On this measure, participants using the GUI met the target criteria of 0% requiring hardcopy. Participants using the command line interface were more likely to require hardcopy documentation in their efforts to perform tasks. Eight of the ten CLI users reached for the manual at least once during the test session, and, on average, CLI users required hardcopy on over half the tasks. While a few of the CLI participants reached for the manual as a matter of course, most first attempted to accomplish their tasks before resorting to the manual.

The following comments reflect the difference:

- CLI user: "I couldn't do anything. I needed to use the documentation and manpages a lot. The time that took was frustrating."
- GUI user: "I could figure out where everything was at. I didn't need to look at the manual. Playing led to learning."

Satisfaction Results. Satisfaction ratings for the GUI exceeded those for the command line interface on all usability attributes and the resulting composite measure, see Figure 3. The GUI ratings exceeded the minimum acceptable criteria (≥ 5 on a 9-point scale) on all attributes and were not statistically different from the target criteria (≥ 7 on a 9-point scale) on five of the seven individual attributes, as well as the composite. Ratings for the command line interface met the minimum acceptable criteria on only the "Power" attribute. Even there, the GUI score exceeded the command line result.

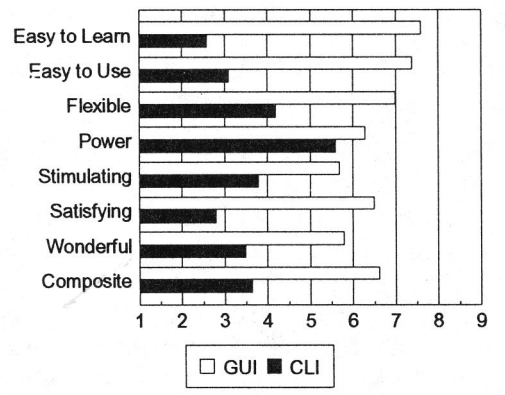

Figure 3. Satisfaction Ratings

These user comments reflect the differences in user perceptions of the two interfaces:

- CLI user: "Working with this was like doing carpentry through a keyhole."
- GUI user: "I believe it [the product] would keep me from having to make a support call."

Implications of the Results. During the design effort, the team found that a sizable volume of networking-related support calls involve helping customers use NetTL, thus significant support cost savings are anticipated. Cost savings are also anticipated from a reduced need for hardcopy documentation. Taken together, these savings were very instrumental in helping the design team justify implementation of their design to management. Harder to measure than the savings to HP but of potentially greater significance are anticipated customer benefits from increased productivity and confidence.

LESSONS LEARNED

Design Process

Multidisciplinary Design Team. Having a multidisciplinary design team helped to make the design process a success. Everyone contributed to the design from their own perspective. The team believes that these different perspectives helped them to develop an interface that met the release criteria. This is because no one view of the product or interface could dominate the design. The members of the team believe that all design teams should be multidisciplinary.

Experience. Having only one person on the team with user-centered design experience does work if that person has the leadership skills and techniques necessary to, in essence, provide on-the-job training in user-centered design. Some of the many challenges this person faces are:

- Frustration that the team has to move slower than desired. This is because the team has to learn as they go.
- Continually keeping the team focused on the process.
- Answering doubts about the process and the value of the process from the team as well as management.

The Process. Product team members are familiar with identifying who their users are, but not defining their users and determining how this information may impact the design. Team members are also relatively unfamiliar with analyzing tasks from a user's perspective. They tend to adopt a product perspective instead. When the design begins and the user profile and task analysis information is used to make design decisions, then the value of the effort becomes clear to the team.

However, even at this stage there can be doubt about the process. Most doubts disappear once the first inspection is complete. The developers are particularly struck by the thoroughness of the inspectors and the amount of criticism and suggestions generated. For perhaps the first time, they realize that what they thought was intuitive isn't necessarily so for their users. The inspection feedback also convinces the team that several iterations of design and evaluation may be needed before the release criteria can be met. In fact, in early inspections, surface structure problems and mental model mismatches can hide fundamental problems with product functionality and/or structure.

Testing Process

By the time the testing of the release criteria happens, the design team is enthusiastically supportive of the process and understands the value of the evaluation. They are eager to see external customers use their design, so it is important to encourage them to observe as many test sessions as possible. During these sessions they can see ways to improve their contribution to the product. However, the test administrator needs to discourage team members from making changes until all test sessions are complete and the data has been analyzed.

There are several areas in the testing process where the test administrator must exert a controlling influence due to the risk of introducing bias. These include: a) interacting with test participants, b) analyzing test data, c) determining whether release criteria have been met, and d) identifying defects and assessing their severity. Because of this, it is important to have an unbiased tester, that is, someone not involved in the product design.

As mentioned previously, two critical defects were uncovered during the user testing. This demonstrates that inspections cannot entirely take the place of user tests. Nonetheless, the effectiveness of usability testing can be enhanced when a product has undergone one or more iterations of usability inspections. Based on experience with both inspected and uninspected products, the numbers of critical and serious defects found during usability testing is consistently less for products that have undergone inspections. Like more and more development teams at HP, the NetTL team determined they would not release their product until all critical and serious defects had been resolved. By finding and fixing defects as early as possible, the NetTL team has been able to freeze the design much earlier and move through the implementation phase more rapidly than is the case with traditional design processes.

Usability tests that validate release criteria can greatly enhance a design team's confidence in their product. Although this type of testing is expensive and should probably not be used to test every design, it does determine whether release criteria have been met. These quantitative results can also be used in many ways, such as to persuade management to support user-centered design or continue funding of the project.

CONCLUSION

Based upon the experience gained through this effort and many others, we believe that following the user-centered design process will result in products that users can use and that as more teams use this process, the value gained will become known. The success of the new NetTL interface has generated new interest and enthusiasm for user-centered design within the local networking products division. Several product teams are now working through the same process with the support of Human Factors. In addition, we are becoming user-centered design advocates by taking the NetTL success story to other local divisions, as well as to the wider HP audience and to upper management. We feel it is important for all user advocates to support this process and to ensure that all teams gain experience in user-centered design.

In conclusion, we believe the following are the keys to the success demonstrated in the NetTL design story:

- Establishing a multidisciplinary team.
- Making a commitment to user-centered design.
- Maintaining that commitment despite periods of doubt and frustration.
- Sharing the role of user advocate.
- Setting aggressive but attainable release criteria.
- Allowing those criteria to help drive design decisions.
- Getting user feedback early and often and using that feedback to refine the design.
- Moving beyond the product perspective to a user perspective.
- Validating that the product has met its release criteria.
- Quantifying anticipated company and user benefits due to improved product usability.

ADDITIONAL READINGS

1. Gould, J.D. How to Design Usable Systems. In M. Helander (Ed.), Handbook of Human-Computer Interaction. North-Holland, New York, 1988, pp. 757-789.

2. Nielsen, J. and Mack, R.L. (Eds.). Usability Inspection Methods. Wiley & Sons, Inc., New York, 1994.

3. Norman, D.A. and Draper, S.W. (Eds.). User Centered System Design. Lawrence Erlbaum Associates, Hillsdale, N.J., 1986.