DRAFT

A Data-Based Console Logger for Mission Operations Team Coordination

Carroll Thronesbery
S&K Technologies, Inc
201 Flint Ridge Plaza, Ste 102
Webster, TX 77598
281-244-5602
c.thronesbery@jsc.nasa.gov

Jane T. Malin
Kenneth Jenks
David Overland
NASA Johnson Space Center
2101 NASA Road 1, ER2
Houston, TX 77058
281-483-2046
jane.t.malin@nasa.gov

Patrick Oliver
Lockheed Martin
2101 NASA Road 1, ER2/LMES
Houston, TX 77058
281-483-2062
patrick.j.oliver@jsc.nasa.gov

Jiajie Zhang
Yang Gong
Tao Zhang
Univ Texas, Houston
P.O. Box 20036
Houston, TX 77225
713-500-3922
jiajie.zhang@uth.tmc.edu

Abstract—Concepts and prototypes[1,2] are discussed for a data-based console logger (D-Logger) to meet new challenges for coordination among flight controllers arising from new exploration mission concepts. The challenges include communication delays, increased crew autonomy, multiple concurrent missions, reduced-size flight support teams that include multidisciplinary flight controllers during quiescent periods, and migrating some flight support activities to flight controller offices. A spiral development approach has been adopted, making simple, but useful functions available early and adding more extensive support later. Evaluations have guided the development of the D-Logger from the beginning and continue to provide valuable user influence about upcoming requirements. D-Logger is part of a suite of tools designed to support future operations personnel and crew. While these tools can be used independently, when used together, they provide yet another level of support by interacting with one another. Recommendations are offered for the development of similar projects.

Table of Contents

1. Introduction

The purpose of console logs is to keep a written record of

[1] 0-7803-8155-6/04/$17.00© 2004 IEEE Correct 2005 number in August.
[2] IEEEAC paper #1543, Version 1, Updated Oct 13, 2004

the events that occur at the console of a flight discipline in the Mission Control Center (MCC) [FCOH, 2003]. They are used for reference during the shift as a reminder of specific times, data values, events, and responses to support subsequent flight decisions by the flight controller. They serve as the basis of the shift handover to maintain mission awareness as new flight controllers assume responsibility for monitoring flight operations. They serve as reference for researching anomalies on previous flights to understand better how those anomalies developed, what actions were taken, and how successful those actions were. Finally, they serve as the basis for a number of a number of reports used to monitor and manage flight support activities.

Traditionally, console logs were handwritten notes written on a tablet kept on the flight controller's console. Increasingly, they are kept in computer documents to improve legibility and availability. D-Logger represents another advancement in keeping console logs. It is web-based, increasing the availability further, allowing incoming flight controllers to maintain mission awareness from their offices before they arrive at MCC. It is also database oriented, supporting better search capability when flight controllers need to research anomalies similar to one they are currently observing. Finally, it also has API interfaces, allowing software agents to make automated log entries of specific routine events. This capability should make it easier for flight controllers to keep complete, accurate records of well-understood telemetry events, freeing them to track more telemetry and to maintain awareness of higher, mission-level events.

Future mission concepts promise to place even more stress on providing flight controllers with adaptable tools that allow them to concentrate more on the missions and less on managing the tools themselves. Exploration mission concepts pose a number of coordination problems for ground support [Ops Concept, 2000]. They include communication delays that make crew autonomy more desirable, supporting multiple concurrent missions, supporting missions with a reduced flight support team of multi-disciplinary flight controllers during quiescent periods, allowing analyses and inputs from remote locations (office, home), and coordination with automated agents that analyze telemetry and report on recognized mission events. Traditionally, voice loops have supported real-time coordination and console logs have supported coordination across shifts within each discipline. Data-based console logs will enable richer written coordination.

In the remainder of the paper, we describe the functions of the Data-Based Logger (D-Logger), the related console support tools with which it can interact, the spiral development model we used to develop it, evaluations of D-Logger, and some recommendations for similar development projects.

2. D-Logger Functions

D-Logger has a number of functions which support consulting the console logs, adding new log entries, searching them, and using them to create reports.

View the Current Console Log

D-Logger has a display of the current console log so the flight controller can readily consult the information already recorded about the current shift. Some data observations and timestamps are important for current decisions about mission events. Reviewing existing notes also helps to ensure that the log notes are a complete record of current console events. The main view of D-Logger is shown in Figure 1. At the very top of the display, the software is identified as the logger. It allows selection of other console tools (Viewport, WorkIT, a workspace tool to organize specialized tasks like anomaly response, and Reports). It shows other major functions of the logger (help, admin functions, and feedback to developers). Finally, the top of the display shows how the user has logged in (tbery has logged into the test0 activity as a BME, or biomedical engineer). The middle of the display is a scrollable area showing the current console log. It consists of a number of entries. The most important parts of an entry are the timestamp (279/15:02) and the text of the entry. These are the columns that appeared on the early paper tablets on which console logs were originally kept. Twenty-four hours of console logs are available in the scrollable, middle section of the display. If the user wishes to see an earlier set of log notes, then a new "end time" can be entered into the blue area at the top of the current logs display. Alternatively, the user can move the time window of the logs display to one day earlier or later by using the arrow buttons near the "end time" specification blank. The bottom of the display is where new log entries are made.

Add an Entry

During times of high activity, it is important for flight controllers to be able to add a log entry quickly. Simply typing text into the text entry box and submitting will cause a new entry to the console log, with a timestamp corresponding to the entry time. The entry area supports a standard set of editing capabilities, along with an "undo-redo" capability and a spellchecker. It also supports inserting hyperlinks, inserting a current timestamp in the text, and putting very specialized icons and text markups to fit the needs of each specific flight controller discipline

using D-Logger. These specialized text markups and icons save entry time for the flight controllers and make the resulting logs more uniform and make subsequent searches more productive. In Figure 1, the BioMedical Engineers (BMEs) have special icons for information called down by the crew, information which has already been read up to the crew, information which still needs to be read up to the crew, important information, information communicated from one flight discipline to another, and "to-do" list items. The flight controller can specify the timestamp (GMT, or Greenwich Mean Time) so that a chronological listing of the log notes reflects the actual sequence of events on the console.

Flight controllers would like to be able to enter log notes with very little effort, yet they would also like the capability to search those log notes later and to use them as the basis for subsequent reports. As more uses are made of the log notes, more information (metadata) needs to be entered to support those uses. The Quick menu, Snippets menu, and the Report Categories area make it easier to enter the information supporting those additional uses.

Report Categories are used to associate individual log entries with a specific heading of a report, like the Console Support heading of the Shift Handover Report. This allows the ReportMaker function to search for these log entries and place them in the report automatically so that there is less manual effort in creating the report.

The Quick menu is for entries that are frequently made by members of a discipline, automatically entering text into the edit area and selecting the appropriate Report Categories. For instance, a common entry is to announce a newly arrived flight controller on shift. The Quick menu selection for this action is illustrated in Figure 2. When the user makes the Console Support selection from the Quick menu, the text shown in the menu is placed in the edit area with bold formatting, and the Report Categories selection of Console Support is automatically selected. The user then replaces the xxx text entries with actual names. At that point, a complete entry has been made in a uniform fashion for consistent appearance in subsequent reports and conducive to subsequent searches. The Quick menu is an example of "knowledge in the world" rather than "knowledge in the head" recommended for software displays by Wright, el. al [2000] and Zhang [1996].

The Snippets menu is similar in concept the to Quick menu with the exception that it is intended to help users manage temporary lists of frequently entered complex text. For instance, Figure 3 illustrates the use of the Snippets for maintaining a list of current document numbers and titles. Flight controllers use specific types documents to communicate with one another across the MCC and to manage issues and anomalies. These document numbers and names appear in the console log when an action is taken on them. By using the Snippets menu, the flight controllers avoid retyping the long document number and name, avoid mistyping the number, and make the log note more amenable to subsequent searches. By selecting the "Modify" option at the bottom of the menu list, the user can enter a new block of text to be added to the menu. New items are added to the top of the list. As new items are added, old items, not used any more, are dropped from the bottom of the list. Snippets are another example of placing knowledge in the computer interface rather than leaving users to manage it on their own. If users need to designate a document which has been discussed recently, they simply need to select the right document name and number from a list rather than looking up the full document number and name from external sources.

Prepare a Shift Handover Report

Near the end of the shift, the outgoing flight controller completes a Shift Handover Report to be used by the incoming flight controller to support continuity of mission awareness across shifts of flight controllers. A big part of the shift handover process involves the console log. The Shift Handover Report provides event-oriented, anomaly-oriented, and issue-oriented views of the console activities in addition to the chronologically oriented view provided by the console log. The ReportMaker function sorts the log messages into report categories for the Handover Report, forming the initial draft. The outgoing flight controller can then add more explanatory text under these categories to fill in any gaps or provide additional information not already appearing in the individual log entries. The format, content, and appearance of the Handover Report can be specified differently for each MCC discipline of flight controllers. Figure 4 shows a partially completed Handover Report. Note that ReportMaker puts the report into Microsoft Word format where the user has full control over its appearance before it is given to the incoming flight controller and archived.

Search Log Entries

The primary reason flight controllers requested a search capability is to retrieve information relevant to current anomalies by finding similar incidents in the past. This enables to them to examine how previous incidents developed, what responses were taken, and how effective those responses were. Storing the log entries in a database expedites these searches. Figure 5 shows the results of a search of log entries of BME notes in the test0 activity, for the string "PPCR5555", a specific document number. Once the log entry of interest is identified, it often becomes

important to view it in the context of the console log where it was entered – to be aware of other events happening at the time of the incident in question. Figure 6 shows the results when the user selects the "View In Context" link. Note that the entry in question is highlighted in the View In Context view to orient the user to the event of interest and its context (some printouts make this highlighting less noticeable than the on-screen display).

Perform Administrative Functions

A number of administrative functions are provided to users of D-Logger. For instance, a group administrator can specify the categories, headings, and appearance of the Shift Handover Report, as well as specify formats for additional reports the group may want to generate from log entries. They can also add new users to their group and define new Quick menu entries for their group. Regular group members do not have the edit privileges for many of the administrative functions but can view them all. In that way, all users know what specifications can be made and how they are currently specified. This supports the group interaction among the discipline members so that they can have the right set of specifications for that group's needs.

Feedback

Feedback has been incorporated into D-Logger to make it easy for users to report bugs, identify awkward functioning or sequencing, and to identify new functions they would like to see. This feature is particularly important to D-Logger since a primary goal of our project is to establish requirements for console support tools so that flight controllers can concentrate on monitoring flight activities and concentrate less on managing their software tools.

3. Related Console Support Tools

While D-Logger can be used independently, it was intended for coordinated use with a number of related tools supporting flight controllers. The tool suite includes the ViewPort, WorkIT, ReportMaker, Notifier, and IBRA [Malin, et. al, 2002].

ViewPort

ViewPort provides an overview into the current data and options offered by the tools in the tool suite. When completed, it should allow a quick glance at the state of console activities for the discipline. Two important parts of this view are an overview of the data being managed by each of the tools and a way to navigate to the tools. A third part, which we have not yet added, is an insight into planned activities and their execution.

WorkIT

WorkIT is a workspace management tool, originally designed to support the organization of multidisciplinary anomaly response teams [Malin et. al, 2002b]. It allows teams to manage tasks, actions, written reports, links to reference materials, and notes during the course of investigating an incident, anomaly, or issue. Like D-Logger, WorkIT is also web based to improve access from the MCC and from flight controller offices, and it is database oriented to enable searches across workspaces to find related issue analyses.

ReportMaker

ReportMaker is a tool for constructing written reports. Its power lies in its ability to communicate with other tools in the suite so that written reports can be generated automatically. The product of ReportMaker can be the final report, normally in Microsoft Word format, or it can be an automatically generated draft, which a person can then edit or append additional information to complete the report. We have seen how ReportMaker can use log entries in D-Logger to make a draft Shift Handover Report. It can be used to generate summary reports of WorkIT workspaces and their current status. The Intelligent Briefing and Response Agent (IBRA) can also generate written reports on events it recognizes by using ReportMaker.

Notifier

The Notifier is used ensure that people are notified appropriately of important events [Martin, et. al, 2003; Schreckenghost, 2002]. The modalities available to the user include the user interface in the tool suite environment, pagers, and email. The way in which a notice is constructed and sent depends on characteristics of the recipient (online vs offline, current role in the project, personal preferences), characteristics of the event, and decisions of the group about how members should be notified of important events. The Notifier is used by other tools in the suite. For instance, IBRA could use the Notifier to alert people about a new event it recognized, or WorkIT can notify someone that they have been assigned a new analysis task for a specific issue.

IBRA

IBRA is a tool for recognizing patterns in the data and taking appropriate actions [Malin, et. al, in preparation]. For instance, IBRA can watch telemetry data and report that the Remote Manipulator System has been returned to its resting position. A report of a nominal state like this would probably appear as a log message automatically entered by an IBRA action. On the other hand, if IBRA were to recognize an anomalous state that needs immediate

attention, its actions might include sending a notice through Notifier, beginning an anomaly report through ReportMaker, and starting a WorkIT workspace for use by an anomaly response team, as well as making a log entry through D-Logger.

4. Spiral Development Model

We are employing a spiral development model to D-Logger and related tools [Boehm, 1988; Thronesbery & Malin, 1998]. This means that we first build the basic functioning of a given tool, get feedback from users, and then design improvements and additional functioning based on the initial usage feedback.

Basic D-Logger Functions

The basic function of keeping a console log includes keeping a record of console events that can be consulted to guide future console decisions, support shift handovers, and maintain archives of those events. The original paper tablets fulfilled these functions in the simplest way, but left flight controllers wanting a bit more support. Keeping that information in word processing documents improved the legibility and accessibility of those logs. However, flight controllers needed a little more functionality to feel they had a complete basic set of console logging support. The most important of these additional functions is a better capability for searching through existing logs. These searches help them to find things like specific observations, settings, and event times, as well as to locate previous anomalies similar to current observations. Consequently, we have included in the basic functioning of D-Logger, the web-based application which keeps log entries in a database to support searches. Since the handover report consists primarily of log notes already entered into the computer, we also included the capability to generate a draft handover report from those log notes, eliminating the need to retype that information. Also, we have added a number of features to make it easier to enter the additional data to support handover reports and effective searches. This helps to make it easier, rather than more difficult, to create a console log with more information. We included the feedback functions to ensure that we collect as much information about requirements as possible. Finally, we included a sizeable complement of user specification tools (admin functions) so that each discipline using D-Logger could tailor it to their specific and changing needs.

Future D-Logger Functions

We have plans for adding more specialized functions to the logger. These plans involve integrating support for tracking things like to-do lists, data uploads, paperwork, and planned console activities into the logger. In this way, as a result of tracking these activities, the flight controller can have log notes automatically entered from the tracking tools (no longer needing to both track and manually report on the activity). A full set of metadata can be added to those logs so that they can be used in creating tables for reports (e.g., a summary in the Shift Handover Report of all the currently open paperwork along with actions taken on it during the last shift). This integration of support for tracking activities and reporting in the console log involves a closer association with work processes employed by the flight controllers as they track these items, but it should be a big step in improving the effectiveness with which flight controllers are supported by their software. An additional challenge is to design new admin functions to allow flight controllers of each discipline to specify activity tracking. This capability would eliminate their need to formally request those capabilities from a software organization.

We also plan to integrate the logger more closely with the other console support tools. For instance, the IBRA agent, which can recognize important telemetry events can automatically enter a log note (identifying itself as the creator of that log note). This would eliminate the need to record those telemetry events manually into the written history of console events. Again, those automatically entered log notes will be tagged so that they can appear in appropriate places in reports like the Shift Handover Report. Similarly, some of the WorkIT activities should be recorded in the console log to report on progress in analyzing related issues and anomalies. Some of these activities should be reported automatically and some should be reported as an option taken by those people working the anomaly. In either case, WorkIT would communicate with D-Logger to enter a note in the console log.

5. Evaluations

Because a major project goal is to derive a good set of requirements for console support tools, evaluation has been an integral part of the project from the outset. We consulted with users concerning requirements before beginning the design process, and we used those requirements in every type of evaluation since then. In addition, we used every opportunity when talking with users to refine those requirements. The sequencing of the evaluations was guided not only by principles discussed in Thronesbery and Malin [1988] of increasing scenario fidelity and complexity, but also from user input on what sequence of evaluations would make them feel comfortable in using D-Logger to support actual missions.

Expert Walkthroughs

The initial evaluation for each function of D-Logger was in the form of an expert walkthrough. For functions that were expected to require innovative user interaction, the initial design was a paper prototype created from a user-centered perspective. The design was made employing the users' statements about their desire for console support, example artifacts they currently use in performing their jobs, and specific data to support one or two common usage scenarios. The expert walkthroughs included experts from software design and database design, as well as those from user interaction design. We modified checklists from Lewis and Rieman [1993] to guide the analysis from the user's cognitive task perspective and to ensure a thorough heuristic evaluation of the general characteristics of the user interaction. When a problem was encountered from one perspective, a discussion would ensue concerning alternate designs or implementation strategies so that the design would be workable from all perspectives. Consequently, we finished the walkthroughs with designs that not only support users, but also can be developed within the scope of project assets.

User Walkthroughs

When we had a working prototype with a full set of basic functions, we performed the first user evaluations. These included four evaluators from a specific discipline we identified as our primary initial user, the biomedical engineers (BMEs). We also included two evaluators from other disciplines so that we could be aware of how D-Logger should be tailored to fit the needs of additional disciplines of flight controllers. Evaluations were performed individually, taking about two hours per evaluation. To provide structure for the evaluation sessions, we prepared a walkthrough scenario to illustrate the logger functions. The session consisted of describing the usage scenario, describing each D-Logger function, and allowing the evaluator to exercise that function. The evaluator was encouraged to provide feedback as the session progressed. Following an approach described by Woods, et. al [1996], we were looking to verify if we had an accurate understanding of the requirements, if we had adopted a reasonable strategy to support those requirements, and if we had implemented those strategies effectively. This approach helps to organize the results of a formative evaluation so they point to more obvious improvements in the evaluated software. In addition, while the users were interacting with the software, they were encouraged to think aloud to give us further insight into their work processes as well as difficulties they may experience with the software. The evaluation sessions were videotaped to identify moments when flight controllers experienced difficulty with D-Logger as well as to ensure that nuances of think-aloud feedback and suggestions for improvement would not be lost. At the end of the session, we had a questionnaire of open-ended questions, beginning with general questions and concluding with questions about specific issues concerning the design.

This evaluation was a formative evaluation, aimed at finding ways of improving the existing prototype so that it would be ready for the next level of evaluation. The evaluation indicated that the performance speed of the system needed improvements before D-Logger could be used to support missions. It also indicated wording changes for menu items and a better set of Quick menu items. The Shift Handover Report had evolved since we last talked with users, so we also got an improved format for that report. In addition, we got new handover report artifacts to match the new format. We also received indications of what new features our users would find most helpful. We have not yet had the opportunity to incorporate those new features, but this advanced notice has helped us to plan for those features and to consider possible approaches for incorporating them.

Observed Trial Usage During Mission Simulations

After we had incorporated some of the lessons learned from the user walkthroughs, D-Logger was ready for an evaluation under more realistic usage conditions. After consulting with users, we chose to evaluate D-Logger during mission simulations, balancing the need for a realistic evaluation and the need to avoid any risk from using an unknown tool during an actual mission. Mission simulations are performed for flight controller training to ensure that they know how to respond to any mission occurrence. Simulations are planned for a specific duration and to exercise a particular type of mission scenario, introducing a number of anomalies not known in advance by the flight controllers. We chose to evaluate during two mission simulations, lasting from six to eight hours. Because shift handovers were an important concern, they arranged to have a shift handover midway through each simulation. We videotaped the use of D-Logger during the mission simulations and encouraged the flight controllers to talk aloud while using the logger when circumstances would allow. We prepared a set of questions aimed at increasing our understanding of the users' tasks, assessing our choice of strategies for supporting those tasks, and assessing the implementation of those support strategies. We also prepared questions to guide our choices for future functions and their design. These questions supported impromptu discussions with the flight controllers during low activity periods during the simulations. At the end of the evaluation session we asked the questions which had not already been addressed during the simulation.

During this evaluation, we received additional information about improving the Shift Handover Report. Their format had again changed again, but most importantly, using the format and automated software functions in a realistic context clarified their requirements. The evaluation identified the need for new Quick menu items; it exposed some new security concerns about read and write privileges; and it indicated the need for a new edit button. The performance speed during this evaluation was sufficient to support a mission. Because of the extended length of the test and its more realistic use conditions, we were able to uncover new bugs. We also found a few new requirements concerning timestamps for simulation exercises. Only the bugs, timestamps, and security concerns needed to be addressed to be ready for the next level of evaluation.

During the course of these discussions, we were able to collect a larger set of artifacts that flight controllers use to perform their console activities. These artifacts have greatly increased our understanding of how new support functions in the logger should be designed.

Free Play User Evaluations

We are ready for unobserved trial use (free play) by flight controllers. In this evaluation, the flight controllers will use D-Logger during simulations, in a flight following mode, or just to exercise any function they are curious about. The common factor in all these exercises is that we will not be there to observe the use of D-Logger. Flight controllers will exercise it in any way necessary to gain the comfort needed to use it during an actual mission. The data planned for this evaluation is a little less direct than that of previous evaluations. We will rely heavily on the online feedback system and an online questionnaire.

Online Feedback

The online feedback function is designed to get user feedback concerning bugs, information about task understanding, our choice of strategies for supporting tasks, and our implementation of those strategies. The feedback also includes users' requests for new functions. A feedback item includes automatically collected information about the usage context when the feedback system was called. It also includes a description of the problem or improvement. Feedback items are emailed to key people on the development team so that they can give the items immediate attention. Feedback items are also maintained in a database so that they can be managed like a bug list or a list of future requirements, depending on the nature of the specific feedback item.

Online Questionnaire

Once the users have had enough experience to form an opinion of D-Logger, they will be asked to complete an online questionnaire. The questionnaire has been fashioned after the standard QUIS questionnaire discussed by Harper, et. al [1997], but with the evaluation items tailored to software which supports maintaining a console log in the MCC. The online questionnaire is the first evaluation we will conduct whose primary objective is a summative evaluation of how well D-Logger supports its users. Evaluations prior to this have been formative, aimed at identifying how D-Logger should be improved and what new functions should be added in the near future. The value of this summative evaluation is to form a baseline so that the level of support of this and subsequent versions of D-Logger can be directly compared.

Observed Trial Use During a Mission

When the free play evaluations have concluded and any needed improvements are made, then D-Logger will be ready for evaluation in the context of supporting an actual mission. After that, D-Logger can be adopted for regular use while research continues on new functions (automated log entries, use of logger by related console support tools).

6. Conclusions and Recommendations

The human-centered approach to software development has helped us to design a logger based on a realistic understanding of the console activities it must support. This understanding also includes the ongoing tasks that serve as the context of logging activities. This understanding of the users' tasks has helped us to identify good strategies for supporting these activities. Preliminary results indicate that we have also arrived at reasonably effective implementations of those strategies. The approach, originally formulated in Thronesbery and Malin [1998], inherits from a number of influences from spiral development [Boehm, 1988], user centered design [Norman & Draper, 1986], contextual design [Beyer & Holtzblatt, 1999], ethnographic approaches [Wixon & Ramey, 1996], to methods of bridging the gap between requirements and implementation [Woods, et. al, 1996]. It has been refined over the course of this general project for designing tools for support of flight controllers and has proven effective in the current software design and evaluation of D-Logger. In addition to refining and confirming the value of this general development approach, D-Logger development has allowed us to confirm a number of specific recommendations for innovative software development projects.

Be Useful Early

The first recommendation is to be useful early, and add to basic functions later. Identifying the core of functionality provides focus to the entire development team and helps to avoid time-consuming, schedule-breaking distractions. Most importantly, it helps to establish the right relationship with users. Early interactions with users are more focused. They can trust your ability to deliver a finished product when you produce a product that is useful. At that point, they can begin to use the innovative product and understand it on new levels as users incorporate it into their work process. The sooner users can begin using the innovative support software, the sooner they can give you informed feedback on how to improve that software.

Integrate Evaluation into All Phases of Development

The next recommendation is to integrate evaluation into all phases of development. This provides continuous user influence on the development process and improves the probability of a useful, usable product. Early phases of evaluation should concentrate on task understanding. Subsequently, focus should move to strategies for supporting tasks. Then, the focus should move to effective implementation of those strategies. When a new area of functionality is added, then this progression of focus from understanding, usefulness, and usability should begin again.

Make Tools Independently Useful

Making tools independently useful is a special case of the first recommendation. For instance, we made the WorkIT workspace tool available first. It is a fully functional tool that does not require any other tool to be useful. Flight controllers are now using WorkIT in its standalone form and are providing feedback on how to improve it. We are in the process of making D-Logger available as an independently functioning tool. Later, users who care to use both can take advantage of integrating the two tools. However, users are not required to have both in order to find either tool useful. This makes it easier for users to add the tool to their suite of support software.

Make Tool Interoperate

Making tools interoperate helps to support the users' overall tasks. This helps users to perform a task once without having to enter the same information multiple times. Ideally, it should help users concentrate on the their tasks without having to concentrate on managing their tools. For instance, a normal function of WorkIT is to close out an issue when the analysis has been completed. If WorkIT allows the user to record the closing of an issue in the console log without any additional effort, then we have made the flight controller's job easier.

Allow Users to Customize Their Software Tools

Finally, we recommend that users have the ability to customize their tools. Many features of D-Logger provide very specific support that applies to only one of about 14 disciplines of flight controllers. However, we have taken special care to allow customization of those parts of the support that are specific to one discipline. For instance, the Quick menu is currently loaded with items that are frequently entered by BMEs. However, a very short session with the admin functions can change the Quick menu items to those that will be useful to another discipline. This not only allows the tool to be tailored to each discipline, it also allows each discipline to tailor the tool to their changing requirements. This allows the tool to remain useful in the face of changes in what is required of the users over time.

Acknowledgement

This work is funded by the Human Centered Computing area of the Intelligent Systems Program managed by NASA. The authors want to thank Mike Shafto for his constant support of this interdisciplinary work. [Jane I need help here.—CT]

References

[1] International Space Station Flight Controller Operations Handbook (JSC-29229, rev DCN006). (October 2003). Houston: Johnson Space Center.

[2] Operations Concept Definition for the Human Exploration of Mars (DV-00-014) 2d ed., May 17, 2000. Houston: Johnson Space Center.

[3] Wright, Peter; Bob Fields; & Michael Harrison. (2000). Analysing Human-Computer Interaction as Distributed Cognition: The Resources Model. Human Computer Interaction 15(1):1-42.

[4] Zhang, J. (1996). A Representational Analysis of Relational Information Displays. International Journal of Human-Computer Studies, Vol. 45, 59-74.

[5] J.T. Malin; Johnson, K.; Molin, A.; Thronesbery, C.; & Schreckenghost, D. (Mar, 2002). Integrated Tools for Mission Operations Teams and Software Agents, IEEE Aerospace Conference.

[6] Malin, J.T.; L. Hicks; D. Overland; C.G. Thronesbery; K. Christoffersen; & R. Chow. (February, 2002). Creating a Team Archive during Fast-Pasced Anomaly Response Activities in Space Shuttle Missions. NASA Technical Report (NASA/TP-2002-210776). Houston: NASA Johnson Space Center.

[7] Martin, C.; D. Schreckenghost; P. Bonasso; D. Kortenkamp; T. Milam; & C. Thronesbery. (2003, March). Aiding Collaboration among Humans and Complex Software Agents. AAAI Spring Symposium. Workshop on Human Interaction with Autonomous Systems in Complex Environments. AAAI Spring Symposium.

[8] Schreckenghost, D., Martin, C., and Thronesbery, C. (2002). Specifying organizational policies and individual preferences for human-software interaction. Proceedings of AAAI 2002 Fall Symposium Workshop on Etiquette for Human-Computer Work. November 2002, (North Falmouth, MA).

[9] Boehm, B. (1988, May). "The Spiral Model of Software Development and Enhancement," IEEE Computer, 21(5), pp.61-72.

[10] Thronesbery, C.G., & J. Malin. (1998, July). Field Guide for Designing Interaction with Intelligent Systems. NASA Technical Memorandum, NASA TM-1998-208470.

[11] Lewis, Clayton; & John Rieman. (1993). Task-Centered User Interface Design: A Practical Introduction. An online book. http://hcibib.org/tcuid/tcuid.pdf

[12] Woods, D., Patterson, E., Corban, J., Watts, J.(1996). Bridging the gap between user-centered intentions and actual design practice. Proceedings of the Human Factors and Ergonomics Society 40th Annual Meeting. Philadelphia, PA.

[131] Harper, B., Slaughter, L., & Norman, K. (1997, November). Questionnaire administration via the WWW: A validation and reliability study for a user satisfaction questionnaire. Paper presented at WebNet 97, Association for the Advancement of Computing in Education, Toronto, Canada. http://www.lap.umd.edu/QUIS/publications/harper1997.pdf

[14] Norman, Don; & Stephen Draper. (1986). User Centered System Design. Hillsdale, NJ: Lawrence Earlbaum..

[15] Beyer, H.; & Holtzblatt, K. (1999). Contextual Design. Interactions, vol 6.1, 32-42.

[16] Wixon, D.; & J. Ramey (Eds.) (1996). Field Methods Casebook for Software Design. New York: John Wiley & Sons.

Biography

Carroll Thronesbery…… ……

A Data-Based Console Logger for Mission Operations Team Coordination

Carroll Thronesbery, Jane T. Malin, Kenneth Jenks, David Overland, Patrick Oliver, Jiajie Zhang, Yang Gong, Tao Zhang

Abstract—Concepts and prototypes are discussed for a data-based console logger (D-Logger) to meet new challenges for coordination among flight controllers arising from new exploration mission concepts. The challenges include communication delays, increased crew autonomy, multiple concurrent missions, reduced-size flight support teams that include multidisciplinary flight controllers during quiescent periods, and migrating some flight support activities to flight controller offices. A spiral development approach has been adopted, making simple but useful functions available early and adding more extensive support later. Evaluations have guided the development of the D-Logger from the beginning and continue to provide valuable user influence about upcoming requirements. D-Logger is part of a suite of tools designed to support future operations personnel and crew. While these tools can be used independently, when used together, they provide yet another level of support by interacting with one another. Recommendations are offered for the development of similar projects.