Interface Design for a Modern Software Ticketing System

Minhui Xie, Mark Tomlinson, Bobby Bodenheimer
Department of Computer Science
Vanderbilt University
Nashville, TN 37235
{minhui.xie, mark.tomlinson}@vanderbilt.edu, bobbyb@vuse.vanderbilt.edu

ABSTRACT

This paper describes issues in the design of IT-centric trouble-ticketing applications. Two prototypes are presented. The first introduces a user-centric, web-accessible thick client interface that seamlessly integrates features most commercial companies offer only as bolt-on, value-added additions. The second prototype is an HTML-designed, calendar-centric ticketing application that provides unique affordances to helpdesk technicians. Expert evaluations show both models are superior to current systems. Implementation issues are discussed.

Categories and Subject Descriptors
H.5.2 [Information Interfaces and Presentation] User Interfaces
Keywords
Trouble-ticketing, XUL, user interface design

1. INTRODUCTION

Trouble-ticketing systems are used in a number of service-oriented industries to quickly and efficiently keep track of jobs and job-related details. Within the IT industry, ticketing systems, or helpdesk applications, form the core of most technical support groups. For example, a simplified usage scenario is:

Dr. Johnson of Fictitious University arrives at the office one morning only to find his computer unable to start up; it just sits there making an intermittent pinging sound. Dr. Johnson calls Adam, a Fictitious IT helpdesk agent. Adam creates a new trouble ticket for Dr. Johnson, records his name and location, and then listens to a description of his computer problems. Adam runs through a basic troubleshooting checklist; when the problem is not resolved, she transfers the details to the open ticket. The ticket is assigned to Julie, a support technician. Julie logs into the ticketing system later in the morning, pulls up Dr. Johnson's ticket, goes to his office, determines that the hard drive is faulty, replaces it, closes Dr. Johnson's ticket, and proceeds to her next ticket.

Unfortunately, there are a number of design problems common to many of the helpdesk packages available today. Most commercial products focus primarily on backend technologies, such as lightweight directory access protocol (LDAP **[7]**), and value-added solutions, such as asset and change management, instead of user-interface design and evolution. Many of these value-added products are clearly bolt-on additions. In a few cases they are from other companies entirely. Almost none integrate cleanly with existing base products.

Given increasing corporate pressure to transition from traditional operating system-dependent applications to web-based counterparts, many vendors simply duplicate their existing interfaces in HTML. Unfortunately, it is not yet possible to do a wholesale dump of complex legacy interfaces to web technologies without a significant redesign to compensate for the different affordances that web technologies bring.

This paper attempts to correct these shortcomings. We conducted ethnographic observations of technicians engaged in IT helpdesk support using standard products. Based on these observations, we formulated four shortcomings of existing interfaces. We then surveyed several helpdesk packages to see how these flaws were addressed. Finding that they were not addressed adequately, we introduce two new ticketing system prototypes. One is a XUL-based **[3]** application designed around users' work and thought flows. It has a thin client-accessible thick client interface with seamless contacts, asset management, and knowledge-base integration. The other prototype is a concept model of a pure HTML interface offering a number of calendar-centric affordances that enhance the ability of technicians to locating their most important tickets.

ACMSE '04, April 3-4, 2004, Huntsville, AL, USA.
Copyright 2004 ACM 1-58113-870-9/04/04…$5.00.

2. RELATED WORK

The basics of good user interface design can be found in many textbooks, e.g. **[9].** Unfortunately, many of those techniques have not found their way into existing ticketing applications. A large listing of helpdesk packages can be found in **[8]**. We evaluated five different commercial and open-source helpdesk packages. Each system demonstrated multiple design problems. For example, Best Practical's *RT* **[1]** is a leading open source ticketing package with an integrated knowledge base. Searching for a knowledge base article while entering a ticket requires manually opening up a second browser window/tab and (possibly) logging into the system again. In addition, the default ticket view is not customizable and shows only the ten highest priority tickets. There is no intuitive address book/ contacts integration. With various minor exceptions, most other vendors follow similar approaches. Even though most commercial packages have various levels of customizability built in, none of the packages we examined allowed modifications that address the issues in this paper without significant rewriting in Perl and/or HTML. As far as we are aware, there are no other published papers that cover these specific topics in this context.

3. SYSTEM DESIGN

3.1 Observations

We observed technicians engaged in IT helpdesk support at Vanderbilt's ITS computing center. Based on our observations and interviews with these technicians and other personnel with significant experience in the IT support field, we obtained the following characteristics of the average support technician: (1) Technicians constantly sort and filter their tickets to locate those which match specific criteria, usually highest priority and closest deadline. Secondary criteria often include basic problem type, location, and user availability. Within those criteria, tickets are usually handled first come, first served. (2) Most tickets are handled within a timely matter, usually within a week after they are created. Thus, most technicians are primarily concerned with tickets created recently. (3) Actual work time spent on tickets varies widely, and most technicians are assigned up to 10-15 tickets per day. These observations are specific to Vanderbilt ITS, but, based on conversations with technicians at other locations, we feel these characteristics hold for the "average" technician. The characteristics can vary widely due to the size of the installation, number of total technicians, and average number of users/computers supported per technician.

Based on these observations and a survey of existing trouble-ticketing systems, we present four areas for improvement in existing helpdesk packages.

3.2 User Optimized

Most IT ticketing system users focus on a basic set of tasks–entering new tickets, identifying the next ticket to work on, and updating/closing those tickets. All of these involve factors such as juggling ticket priorities, due dates, etc. After observing a number of technicians at work, we streamlined the interface to enhance the features most used and hide those that are not common.

3.3 Integrated Value-Added Services

Some helpdesk packages are highly modularized, enabling vendors to charge separately for each particular piece. Other vendors charge extra for "advanced" technologies–asset management, change and configuration management, and the like. In almost every case, these "bolt-on" packages do not integrate well with the base application and fail to take full advantage of any associated affordances.

In particular, asset management and knowledge-base systems are critical components of the IT support infrastructure and should be seamlessly and transparently integrated.

3.4 Information Mapping

Information mapping **[2]** is a scientific methodology used to divide and label information for easy comprehension, use, and recall. In the context of this paper, it is a way of thinking and communicating whereby technicians approach content with a set of systematic principles and techniques to ensure that the content can be readily used.

3.5 Interface Separation

The previous generation of ticketing systems used system-dependent, windowing-based interfaces. As the Internet evolved, most helpdesk vendors migrated their systems to HTML. Unfortunately, many of these products were not redesigned to take full advantage of new and different affordances provided by the Internet and its associated technologies.

Some ticketing systems apply this one-interface-fits-all mentality to the customer as well, presenting a single-unified interface used by technicians, agents, and customers. A few interfaces are stretched almost to the breaking point to make them functional on both screen-, resolution-, color-, and input-challenged hand-held devices and standard PCs.

In contrast, the prototypes outlined in this paper are single-purpose models designed expressly for IT technicians and agents and the tasks most common to them. These prototypes take full advantage of the underlying programming medium to deliver a fast, responsive, and efficient interface. In addition, we take the idea of interface separation further to include "supervisor" functions such as charting and reporting. While nice, these features are for the most part useless to agents and technicians and are not included in our designs.

4. PROTOTYPE DESIGNS

4.1 Helpxulla

Our first prototype, Helpxulla, takes full advantage of e*x*tensible *u*ser interface *l*anguage (XUL) to create a thin client ticketing system with the power of a thick client interface. XUL **[3]**, originally developed by Netscape Communications Corporation for use with their next generation web browser (now Mozilla), is a cross-platform XML-based user interface language with a complete set of graphical widgets and data access capabilities. By using XUL as the base for this prototype, we get the full power and graphical toolkit of traditional-style applications, the web-accessibility that is so desirable in modern applications, and the affordances of Internet technologies all combined into one package.

Given the multitasking nature of today's computer systems, one of the more archaic limitations of every web-based ticketing package (we are aware of) is its single-tasking nature. None of the systems we surveyed supports multiple, concurrently displayed tickets, despite the number of uses for such a feature. To support concurrent ticket display and multi-task accessibility, *Helpxulla* has a tabbed interface. Everything – new tickets, open tickets, searches, search results, and knowledge base articles – can be opened simultaneously in multiple tabs **(see Figures 1, 2)**. Additionally, the content of the default "home" page is completely specifiable. Phone agents can set the default screen to show a new ticket. Support technicians can choose to see the current results of any search. The choice is up to the user.

The layout of the "new ticket" portion of *Helpxulla* is completely engineered around how agents and technicians enter tickets. Contact information is accessible in-line via an auto-complete drop-down box. Rather than typing the customer's name and clicking a lookup button only to find out that the name was misspelled, as the agent types in the customer's name, the backend data source is automatically queried; matching names followed by some type of identifying information (e.g., department name) are displayed. The agent need only select the correct name to populate basic location/contact fields. The entire operation takes only a few seconds and can be done seamlessly while the customer is beginning to describe his problem. Heuristic spelling algorithms could, of course, be included if desirable.

Along with the customer's contact information, a short summary of the customer's computer configuration also appears in a collapsible, right-oriented sidebar. Full system details are available as an additional ticket-specific tab, if necessary.

As the user's contact and system information is populating, the agent simply continues down the page entering the information that the customer is describing– error message, details, and a short summary of the problem. As the agent types, a knowledge-base system scans for key words in the background. As soon as possible knowledge base entries have been located, they are shown via an unobtrusive panel at the bottom of the user's screen. Once the agent sees a likely knowledge base article, they can open it in another tab. Like the autocomplete field for the customer's name, there is no need for an agent to explicitly issue a knowledge base query.

Once a ticket is in the system, it is the system's job to ensure that technicians can easily find it. Therefore, the default view for support technicians is the result of a user-defined search. A convenient panel, located in the right sidebar **(see Figures 1, 2)**, lists the number of various types of tickets assigned to the technician. Clicking categories immediately initiates a new search for the relevant tickets and pulls up the results in a separate tab. In addition, the most common search fields form the basis for a complete search screen with integrated filtering.

Once a ticket is opened in a tab, less immediately important information, like contact information and system summary, are displayed in a hideable, resizable right-oriented content panel. The majority of screen space is taken up by the most important data – the user's error message and problem details. Additionally, the latest, most relevant comment is displayed along with the ticket; older comments are available via a separate tab within the main ticket.

Several other features are also useful. Commonly used functions are available in a command bar down the left-hand side of the screen instead of as (potentially nested) menu options. Each of the auxiliary features, such as the command bar and the various content panels, are collapsible to maximize ticket data on lower-resolution displays. Finally, there is a "clone ticket" function that quickly duplicates the current ticket.

4.2 Helplendar

Our second prototype, *Helplendar* (Help Calendar), is a departure from traditional ticketing system interfaces. *Helplendar* particularly emphasizes the navigation of tickets. It addresses a fundamental issue pointed out by most technicians we spoke with: how to effectively locate, manage, and solve multiple calls in different locations, each with unique deadlines. *Helplendar* quickly provides technicians with answers to basic questions such as, "What is the overall situation today" and "How many important tickets are left and what is the status of each of them?"

This calendar-centric interface was inspired by the Treemap **[4]** visualization technique, which provides a compact and elegant means of displaying attributes of a large number of objects **[5]**. The display screen is divided into two portions: a calendar portion on the left and a ticket portion on the right **(see Figure 3, 4)**. The calendar portion displays an entire month at the top (View By Month) and an entire week in the remaining space (View By Week). Technicians can retrieve the tickets in one whole week or up to 100 tickets by clicking a specific day of the month on the calendar. Each ticket assigned to a specific technician will be represented by a visual marker and shown on the week/day of calendar. Closed or resolved tickets are not be shown by default. The tickets being viewed can be arranged in various ways customizable by the technician, e.g. by priority, deadline, chronological order, and service area. The ticket's status and its priority are color-coded so that a technician can easily see how many new or urgent tickets he has pending. Hovering over a color-coded button will trigger a small overlay showing more detail on the ticket in question. Some of the ticket's related information, such as service area (building name) and time of creation, is customizable and can be directly displayed on a button. Technicians can then easily schedule their tasks or arrange their campus trips. Moreover, the balloon feature allows the calendar portion to be hidden temporarily if more space for the ticket's data is desired.

The ticket area **(Figure 3, 4)** is used to process the tickets and implement more of the functionality covered by *Helpxulla*. Technicians can switch between the tickets by clicking on the ticket's visual markers on the calendar portion. This feature provides similar functionality to *Helpxulla's* tabbed interfaces. In *Helplendar*, the content information of a target ticket can be quickly singled out from a large number of individual tickets.

We assume that a technician might want to simultaneously display characteristics (priority, due date, problem category, etc.) of 100 opened tickets on a 19-inch display with a resolution of 1024 x 1280 pixels (approximate 1.3 million). With 10% of screen space already utilized, there are more than 1,000 pixels allotted per ticket. A visual marker with 1,000 pixels (40x25 color-coded button) is sufficient for providing the visualization cues related to the ticket content information and is easily selectable using a standard mouse or TAB key.

Helplendar is a pure-HTML interface. This concept model is browser-based and can be easily accessed from a computer (Windows/Unix PC, Macintosh, Sun workstation, etc.) that has a web browser, such as Internet Explorer (5.x, 6.x), Netscape (6.x, 7.x), Mozilla, and Opera (7.x). Meanwhile, *Helplendar* is completely adaptable in regard to our first prototype, *Helpxulla*, which emphasizes ticket processing.

5. EVALUATION

To evaluate our prototypes, we contacted eight expert users in this field. All of them had at least two years experience working with various trouble-ticketing systems, and half had between ten and twenty years experience. Our evaluation protocol consisted of an introductory tutorial session after which the expert user completed several usage scenarios. Each expert used both prototypes. Half of them used *Helplendar* first and the other half used *Helpxulla* first. Each expert was then asked to list the helpdesk application with which they were most familiar and fill out a questionnaire comparing both prototypes to that system. The questionnaire was based on a five-point Likert scale. An answer of three signified that a prototype was equivalent to the user's standard ticketing system in a particular area, and five signified that the prototype was much better. Because the commercial systems were installed and working, it was not possible to conduct a between-interfaces comparison. Thus, for each response we calculated the mean and standard deviation.

5.1 Helpxulla

The results were encouraging for the *Helpxulla* prototype **(Table 1)**. Our survey contained twenty-one questions for this prototype. Only the most important are reported here, but we note that no answer for any question implied that the expert was more satisfied with his or her existing ticketing package. When asked how useful the tabbed interface was (question 1), the response was very positive with a mean of 4.75, 0.46 standard deviation. When asked how useful the dynamic knowledge base (question 2) and right content panels (question 3) were, the means were 4.13 and 4.5, respectively, with standard deviations of 0.64 and 0.76. When asked about the layout and functionality of the various screens (questions 4 and 5), the mean responses were 4.30 and 4.25 with standard deviations of 0.46 and 0.89, respectively. The mean response to the layout of the search results screen was 4.25. The name auto-complete box (question 6) was also well liked with a mean of 4.63, standard deviation of 0.52.

In general, most of the experts were satisfied with the usability, power, and flexibility of this interface, as means for these questions were approximately 4.15 and standard deviations approximately 0.65. Of some concern was the response to a question asking the experts to rate the power of this interface (question 8). The mean response was 3.50 with a standard deviation of 0.93. It may be that some of the experts had a few of the more advanced features which we have not yet considered, such as ticket linking and service-level agreement support, in mind.

5.2 Helplendar

The results were similarly positive for the *Helplendar* prototype **(Table 1)**. None of the expert reviewers had seen a similar design before (question 1). Out of the twelve questions asked, all were favorable; only the most important are reported here. When asked about the helpfulness of the color-coded button (visual marker) or pop-up overlay in providing ticket-related information (questions 2 and 3), the response was very positive with means of 4.88 and 4.57 and standard deviations of 0.35 and 0.79, respectively. When asked how easy the interface was to use (question 4), the mean response was 4.63, with a standard deviation of 0.74. When asked about the interface's potential to change the agent's view of their data (question 5), the response was surprisingly encouraging with a mean of 4.0 and a standard deviation of 0.82. This result also suggests that a possibly steeper learning curve requiring users to spend some time getting used to this interface.

Some experts we showed this design to believed it would save the technician a considerable amount of time when finding and navigating between tickets. One user went so far as to claim that it had the potential to fundamentally change the agent's view of ticketing data.

6. CONCLUSION

This paper presented improved prototypes for IT trouble-ticketing applications. We ethnographically observed technicians actively engaged in helpdesk issues and solving problems, and determined primary issues that a helpdesk application should address. We then analyzed a number of existing trouble-ticket/helpdesk packages to evaluate how well they met the needs of these users. Their interfaces were found to have usability problems and be poor HTML-duplicates of prior legacy versions. These interfaces also had poor integration of critical ticketing features such as knowledge bases and asset management. To resolve these problems, we introduced two new ticketing models. The first was a XUL-based thin client with the power and flexibility of a thick client interface. The second was a concept model of a calendar-centric design that allowed quickly locating the most important tickets. Expert reviews showed that both prototypes were more useful and flexible than existing helpdesk implementations.

Based on comments from the expert reviewers, there are a number of possible extensions we wish to pursue in the future. One reviewer pointed out that asset data had more affordances than we implemented. He was particularly interested in the ability to separate asset data from ticket data and search it. A number of reviewers also pointed out some more advanced features, such as ticket linking, which were not implemented. Finally, there has been much research done on the use of sliders in database searches **[6]**. We would like to see if it is possible to reduce the search screen and dynamically manipulate the search results.

7. ACKNOWLEDGMENTS

We are grateful to the technicians who allowed us to observe them and to the expert users who reviewed these prototypes and provided very valuable feedback. We thank the anonymous reviewers for their helpful comments.

8. REFERENCES

[1] RT: Request Tracker. http://www.bestpractical.com/rt/

[2] Horn, R.E. Mapping Hypertext: The Analysis, Organization, and Display of Knowledge for the Next Generation of On-Line Text and Graphics. Lexington, MA: The Lexington Institute, 1989.

[3] XUL 1.0. http://www.mozilla.org/projects/xul/

[4] B. Johnson and B. Shneiderman. Treemaps: a space-filling approach to the visualization of hierarchical information structures. In Proc. of the 2nd International IEEE Visualization Conference, pages 284–291, October 1991.

[5] Wattenberg, M. 1999. Visualizing the Stock Market, CHI99, Extended Abstracts.

[6] KEIM, D. A., AND KRIEGEL, H.-P. Visdb: Database exploration using multidimensional visualization. In Readings in Information Visualization: Using Vision to Think, S. K. Card, J. Mackinlay, and B. Shneiderman, Eds. Academic Press, San Diego, 1999, pp. 126-139.

[7] http://www.openldap.org.

[8] http://dmoz.org/Computers/Software/Help_Desk/Browser_Based/

[9] B. Shneiderman, *Designing the User Interface*, 3rd ed., Addison-Wesley, Reading, Mass., 1998.

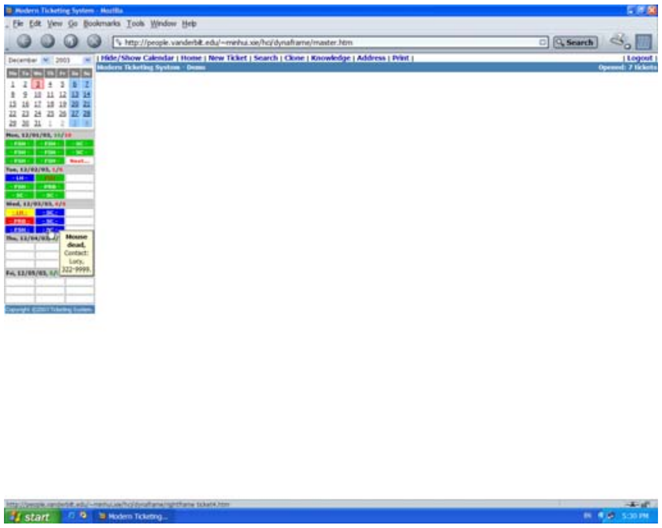
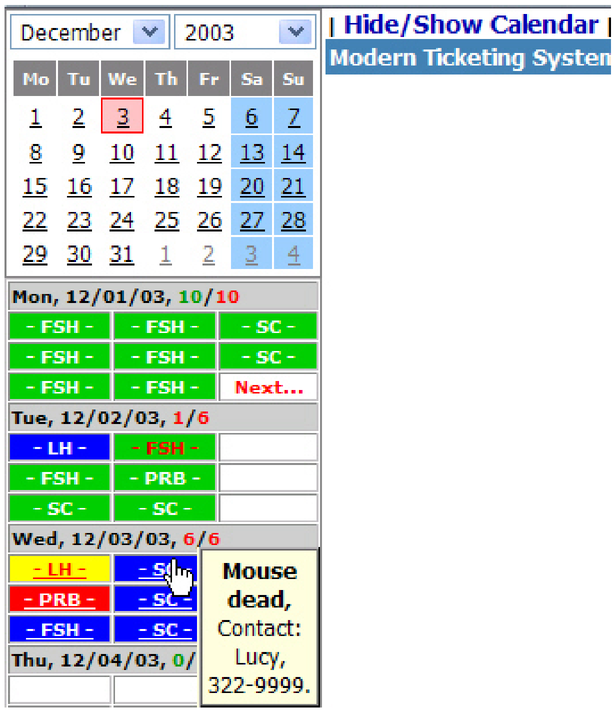


Figure 3: Helplendar – Calendar-Centric Interface



Mouse dead, Contact: Lucy, 322-9999.

Figure 4: Helplendar – Zoom

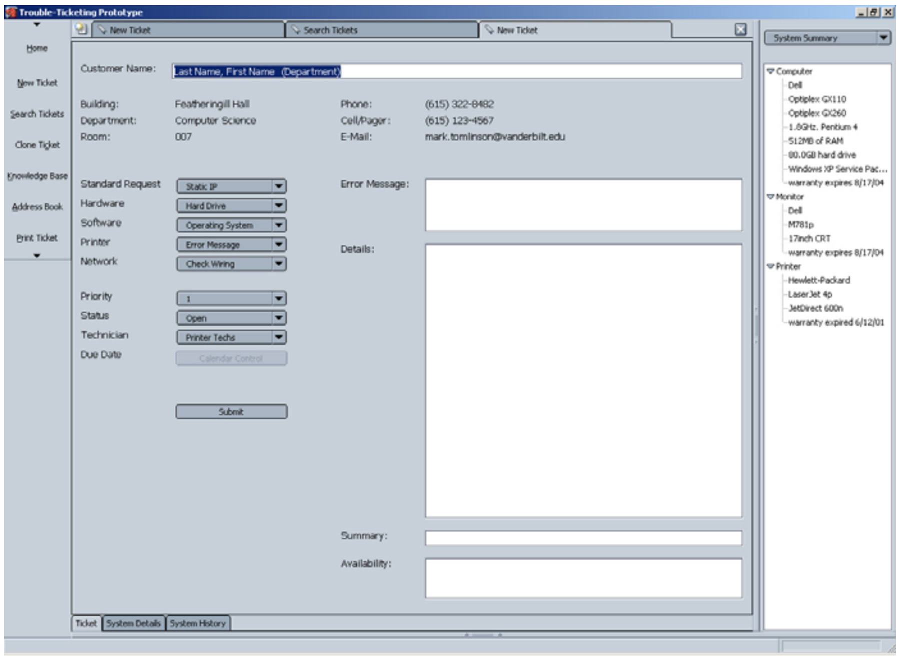
Results of Expert Reviews: Helpxulla

Question	1	2	3	4	5	6	7	8
MEAN	4.75	4.13	4.50	4.30	4.25	4.63	4.15	3.50
STDEV	0.46	0.64	0.76	0.89	0.89	0.52	0.65	0.93

Results of Expert Reviews: Helplendar

Question	1	2	3	4	5
MEAN	-	4.88	4.57	4.63	4.0
STDEV	-	0.35	0.79	0.74	0.82

Table 1: Results of Expert Reviews



Customer Name: Last Name, First Name (Department)

Building: Featheringill Hall
Department: Computer Science
Room: 007

Phone: (615) 322-8482
Cell/Pager: (615) 123-4567
E-Mail: mark.tomlinson@vanderbilt.edu

Standard Request: Static IP
Hardware: Hard Drive
Software: Operating System
Printer: Error Message
Network: Check Wiring

Priority: 1
Status: Open
Technician: Printer Techs
Due Date: Calendar Control

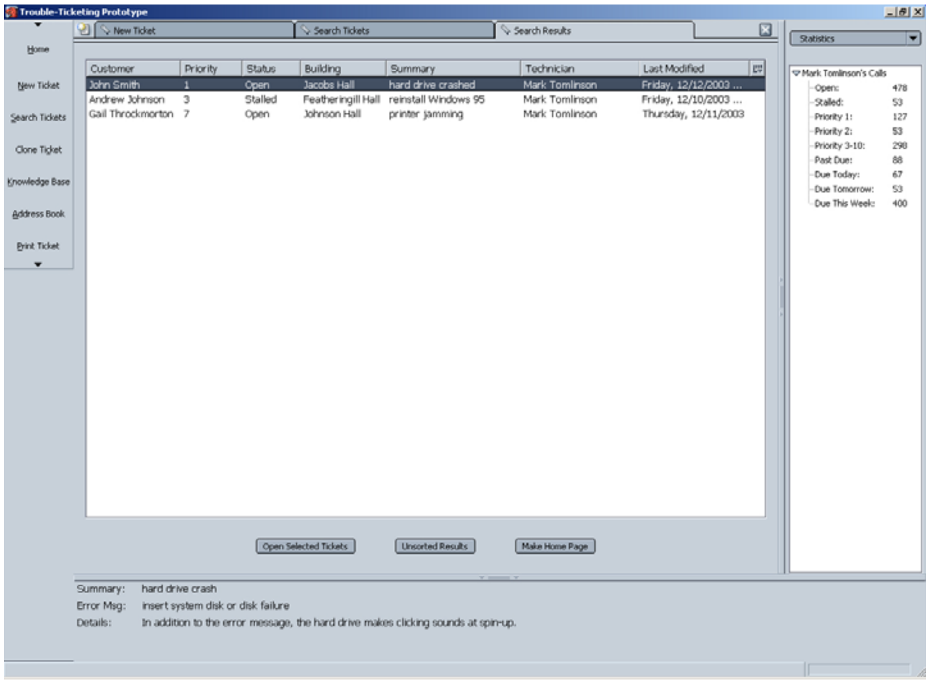
Error Message:
Details:
Summary:
Availability:

Computer
- Dell
- Optiplex GX110
- Optiplex GX260
- 1.8GHz. Pentium 4
- 512MB of RAM
- 80.0GB hard drive
- Windows XP Service Pac...
- warranty expires 8/17/04

Monitor
- Dell
- M781p
- 17inch CRT
- warranty expires 8/17/04

Printer
- Hewlett-Packard
- LaserJet 4p
- JetDirect 600n
- warranty expired 6/12/01

Figure 1: Helpxulla – Create a New Ticket



Customer	Priority	Status	Building	Summary	Technician	Last Modified
John Smith	1	Open	Jacobs Hall	hard drive crashed	Mark Tomlinson	Friday, 12/12/2003 ...
Andrew Johnson	3	Stalled	Featheringill Hall	reinstall Windows 95	Mark Tomlinson	Friday, 12/10/2003 ...
Gail Throckmorton	7	Open	Johnson Hall	printer jamming	Mark Tomlinson	Thursday, 12/11/2003

Mark Tomlinson's Calls

Open:	478
Stalled:	53
Priority 1:	127
Priority 2:	53
Priority 3-10:	298
Past Due:	88
Due Today:	67
Due Tomorrow:	53
Due This Week:	400

Summary: hard drive crash
Error Msg: insert system disk or disk failure
Details: In addition to the error message, the hard drive makes clicking sounds at spin-up.

Figure 2: Helpxulla – Search a Ticket