Usability Testing: A Software Engineering Perspective

Ajay Bandi
Department of Computer Science and Engineering
Mississippi State University
Mississippi State, Mississippi 39762
ab1370@msstate.edu

Phil Heeler
Department of Mathematics, Computer Science and Information Systems
Northwest Missouri State University
Maryville, Missouri 64468
pheeler@nwmissouri.edu

***Abstract*—Background: Software development methodologies adapt customer collaboration in the software development. Because of this, end users of the software participate in the usability testing to give their feedback to the analysts.**

Objective: The goal of this paper is to explore the design issues of the usability testing from the user's perspective.

Method: We followed a systematic approach from the software engineering perspective in preparing the artifacts and conducting the usability testing by defining suitable metrics in a case study.

Results: We identified several design issues while performing the usability testing. Metric values and hit-rates of design issues are presented. We also highlighted our lessons learned and recommendations based on our study.

Limitations: Our case study is medium sized and results may not be applicable to the industry applications. However, the lessons learned and recommendations from this study are applicable.

Conclusion: Users cannot finish complex tasks with no unanswered questions and the identified design issues help in improving user interfaces.

Keywords: usability testing, human computer interaction, software engineering, user interfaces, case study, human subjects, users, design issues, tasks, heuristic evaluation, lessons learned, recommendations

I. Introduction

The purpose of usability testing is to evaluate user interfaces and to ensure the quality of the system [1]. It helps in identifying the design issues from the user perspective. In safety critical systems and other government projects usability testing is mandatory in developing the software. This benefits both the users and the organization involved in it. In agile software development methodology customer collaboration is one of the important aspects while developing the software [3], [6]. Involving stakeholders and end users of the project in the usability testing helps to identify design issues and correct them before the actual implementation of the system.

A. What is usability testing?

Usability testing is a technique to explore the usability issues of a software product using prototypes and necessary documentation with appropriate users [5], [7]. The purpose of usability testing is to find the design issues and correct those issues. The functionality of the system is not required to perform the usability testing. The user in the usability testing must be a member of the intended stakeholders of the project with the necessary domain knowledge and IT background. The procedure and different roles in the usability testing are explained in section III.

B. Design issues in usability testing

Alshamari and Mayhew [2] listed several issues in usability testing including designing tasks, prototypes, identifying appropriate users, and the number of users. In our paper, we focused on the design issues of designing user interfaces. We followed Lauesen's [4] software engineering perspective in designing the tasks, prototypes and reporting the usability problems in a case study. We chose this approach for better understanding of the design issues by preparing different artifacts in a systematic way. We investigated the design issues in the usability testing from the user's perspective.

The next section explains the usability testing approach. Section III describes the details of case study, section IV presents results and its analysis. In addition it highlights the lessons learned and recommendations to software engineering practitioners. Section V discusses threats to validity of the study and section VI presents the conclusions and future work.

II. Usability Testing Approach

Usability testing is not a demonstration of the system. There is no need to have the actual functional of the system.The goal is not to prove that the system is easy to use, and not to find bugs [4]. Avoiding these misconceptions, we followed a systematic usability testing approach to build user interfaces and get the feedback from the users. The following are the steps taken to design user interfaces, to conduct usability testing, and to evaluate the results of the usability testing.

- *Mental Model:* Mental models are the high level conceptual models. A conceptual model is the interaction between the entities represented by boxes and lines. These models are drawn after studying the given case and helps in designing the data model.

- *E-R Diagram:* An entity relationship diagram is helpful to acquire and retrieve necessary data items while designing user interfaces.
- *Task Description:* A task description helps a user to understand the task in some detail. Task descriptions are needed to ensure that the user interface can support the tasks in both simple and complex cases. We followed a structured task description template [4].
- *Virtual Windows (VW)*: A virtual window is a user-oriented presentation of data. Virtual windows are the prototypes created to ensure that all data is visible either on a paper or in the electronic form to the user.
- *Function mini-specifications:* Specifications add required functions to the virtual windows to produce the full prototype by identifying necessary semantic, search, and data entry functions. Specifications can be included by identifying necessary functions to navigate between screens.
- *CREDO matrix:* The CREDO matrix is used to compare the data model and VWs. The columns in the matrix represent the data and rows represent virtual windows. Each cell contains the characters listed below. This helps in finding the missing functions.
    - 'C' for creation of the entity via VW
    - 'R' for reading all attributes via VW or "r" if only some attributes
    - 'E' for editing all attributes or "e" for only some attributes
    - 'D' for deleting through VW
    - 'O' for overview of entities or searching through VW
- *Defect List:* A defect list is the items with the problems identified by the users. In this paper, a defect list refers to design issues.
- *Heuristic Evaluation:* Heuristic evaluation means an evaluator looks at the user interfaces and identifies potential problems. Some evaluators may not have the domain knowledge. Nielsen [8] proposed ten heuristic rules and guidelines to guide the designer during the design process and to help evaluators identify problems in the user interface.

For example, the entity relationship diagram is shown in Figure 1. Task descriptions, function mini-specifications, CREDO matrix are shown in Tables I, II, III respectively. Figures 2, 3, 4, and 5 show the virtual windows and Figures 6 and 7 show confirmation and error messages.

TABLE I. TASKS DESCRIPTION

Task items	Description
T1	**Create a new product**
Start:	An idea for a product is formed.
End:	All product details are decided.
Frequency:	Whenever Bubba decides to come up with a new product, usually dictated by market trends.
Difficulty:	Availability of ingredients for the product may be an issue.
Steps:	
1.	Decide product name.
2.	Pick ingredients.
3.	If it is a new ingredient not used in existing products, then Bubba will find a new or existing vendor for it.
4.	Set the price of the product.
5.	Record product details.
6.	Authorize production.
T2	**View product information**
Start:	When Bubba wants to review the product information.
End:	After the required information (price, product description, ingredients, product manager, etc.) is viewed or retrieved (taking printouts).
Frequency:	Whenever Bubba wants to see an individual product's information.
Difficulty:	Too much information may be displayed on the screen, when viewing many products simultaneously.
Steps:	
1.	Choose the product for which the information is required.
2.	Take a printout if necessary.
T3	**Hire a new product manager**
Start:	When Bubba wants to assign a manager to a product, or hire a new manager to manage a product.
End:	When Bubba assigns the product manager for a product, or hires a new manager.
Frequency:	Not a very frequent task depends on creation of a new product.
Difficulty:	Too many products can be assigned to a single manager.
Steps:	
1.	Generate a new ID for a new product manager, or choose an existing manager.
2.	Store contact details (address, phone number, email etc.) for a newly hired manager.
3.	Assign product to the manager.
T4	**View product manager information**
Start:	When Bubba wants to contact a product manager, or to view performance of the product manager.
End:	After the desired information is viewed or retrieved (taking printouts).
Frequency:	Whenever Bubba decides to contact or view a managers performance profile, which is quite regular.
Difficulty:	
Steps:	
1.	Choose which product manager's information to view.
2.	Select whether to view contact information, performance or both.
3.	Send a message to a product manager.
T4	**View vendor information**
Start:	When Bubba wants to view vendor details, or place orders for ingredients.
End:	After the desired information (contact details, ingredients supplied etc.) is viewed.
Frequency:	Whenever ingredient stock needs to be replenished which is a monthly or weekly process.
Difficulty:	
Steps:	
1.	Choose the vendor, whose information is to be viewed and take printouts if necessary.
2.	Send a message to the vendor.
T6	**Stop the production of a product**
Start:	When Bubba wants to cease production of a particular product or products.
End:	Production of the product is stopped.
Frequency:	Occasionally, when a product is not very popular.
Difficulty:	
Steps:	
1.	Select the product to stop production.
2.	Send message to product manager to stop production.
3.	Send message to vendor to stop supply of ingredients which is unique to the product.

III. CASE STUDY

We start the case study by defining the goal, research questions, and the metrics that we collected. Next we describe the criteria for case and subject selection. The last part of this section explains the procedure for data collection in usability testing. To conduct and report our case study, we followed the guidelines in reporting case study research in empirical software engineering [9].

TABLE II. FUNCTION MINI-SPECIFICATIONS

vwCreateProduct This window prompts the user to enter the name, description, ingredients and price, and then select the product manager. $assignProductManager$: Allows the user to select the product manager by displaying the existing managers names in a drop-down list box. The user selects the manager by clicking on the name from a drop-down list box. $resetFields$: Resets the input fields. $createProduct$: Stores the information given by the user about the new product in permanent data store.
vwViewProductInformation This window displays the information about various products, individually. The user selects the product name from the drop down listbox and clicks OK. When the user clicks OK the system displays the product name, product manager, necessary ingredients and price of the product.
vwStopProductionOfProduct This window allows the user to stop production of a product, by selecting the product from the products currently being produced. $stopProduction$: Sets the status flag of the product to reflect its discontinued production state. $notifyManager$: Allows the user to send an email to the concerned product manager about stopping the production of the product.
vwHireProductManager This window prompts the user to enter the name, contact details and assign the manager to a product. $resetFields$: Resets the input fields. $hireManager$: Stores the name, contact details, the list of products managed by the manager in a permanent data store.
vwViewProductManagerInformation This window displays the contact details, and other information about the product manager, individually. The user selects a product manager from a list and clicks 'View', which then signals the system to display the corresponding manager's details.
vwViewVendorInformation This window displays the contact details, and other information of the product manager, individually. The user selects a vendor from a list and clicks View, which then signals the system to display the corresponding vendors details.

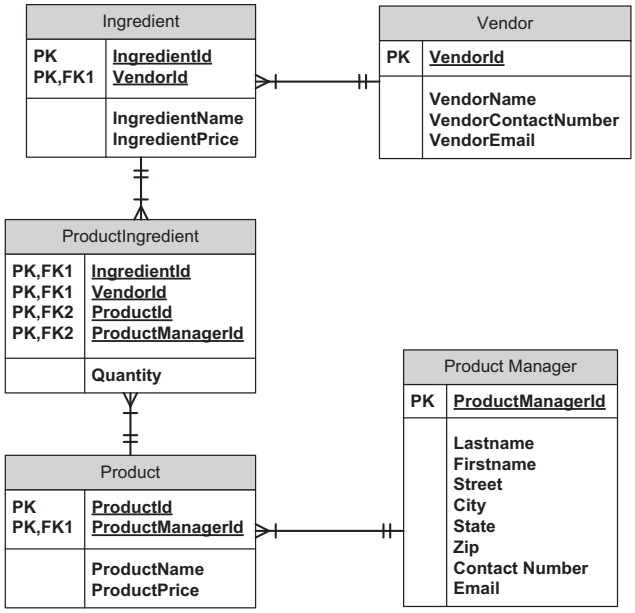

Fig. 1. Entity relationship diagram

TABLE III. CREDO MATRIX

Entity / VWs	Product Manager	Product	Product Ingredient	Ingredient	Vendor
Create product		C			
View product manager information	RO	r	r	r	
View product information		RO	R	R	
Hire new product manager	CE				
Stop product production		D			
View vendor information					RO
Missing functions	D	E	CEDO	CEDO	CED

Fig. 2. Virtual windows for login and main menu.

A. Research questions

Our goal is to investigate the design issues of prototypes from the user's perspective. This goal is further divided into two sub-questions:

RQ1: Can users complete key tasks with no un-answered questions?

RQ2: What design issues do users have when completing tasks using prototypes?

B. Metrics

1) *Time taken to complete a given task:* The metric itself is self-explanatory and the unit for time is measured in seconds.

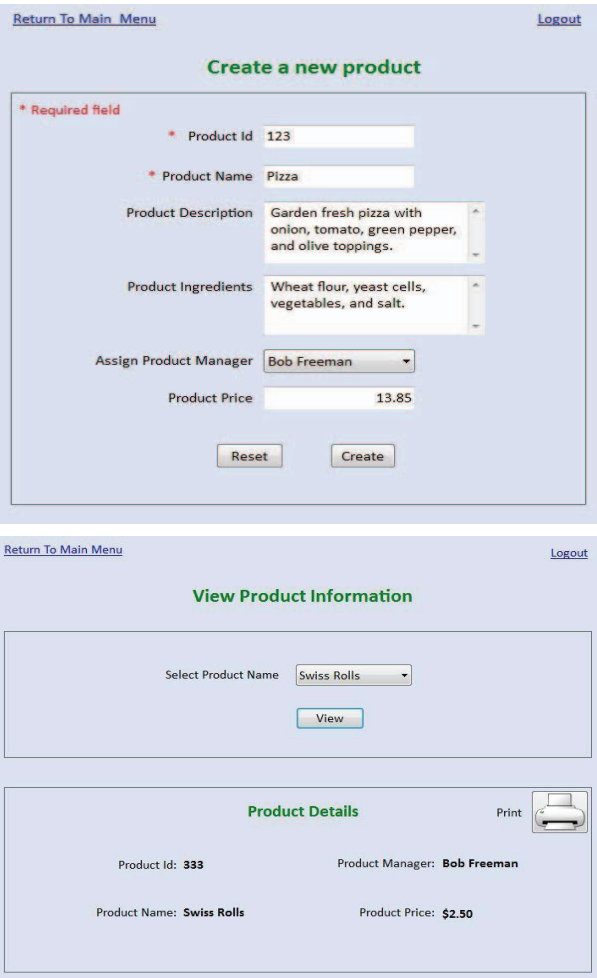

Fig. 3. Virtual windows for create product and view product information.

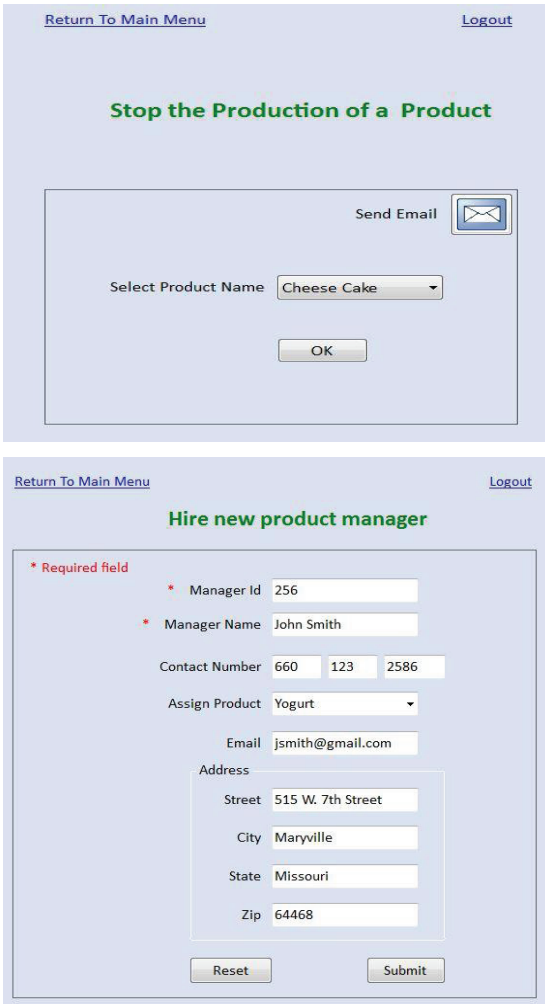

Fig. 4. VWs for stop the production of product and hire new manager.

2) *Number of design issues:* Number of usability problems encountered by the user upon finishing the task.

C. Case Selection

We selected, "Bubba's Healthy Snack Project", a medium-sized academic case study that can be completed in a trimester as a part of our curriculum. The description of the case study is given below. The other deliverables from the case study are discussed in the rest of the section.

Description: Bubba's Healthy Snack Company sells a wide variety of tasty, but healthy, treats. Each of the treats contains several ingredients, each of which is hand-picked for quality from a variety of reputable vendors. Bubba is meticulous about quality, and once she finds a vendor that supplies the highest-quality ingredient, she always buys that ingredient from that same vendor. In some cases, a vendor may have more than one ingredient that meets Bubba's strict standards. Bubba has several product managers who work with her, and each of the product managers is responsible for more than one product. Bubba is convinced that she has assigned the best manager to the most appropriate product, so no product is ever managed by more than one product manager.

D. Subject Selection

The selection criteria of users to perform usability testing is that the user profile should match the intended future user. In other words, domain knowledge of the given case study and Information Technology (IT) knowledge are the requirements to select the users. We selected ten graduate students with IT background and explained to them about the snack project using Power-point slides to acquire the domain knowledge of the snack project.

E. Data Collection

We followed a systematic procedure to conduct usability testing after creating the necessary deliverables. The four different roles in the usability testing are user, facilitator, log-keeper and computer.

1) **User** – Subject who performs the given task.
2) **Facilitator** – Test team member who has the main contact with the test user.
3) **Log-keeper** – Test team member who records the time to finish a task and issues encountered by the user.

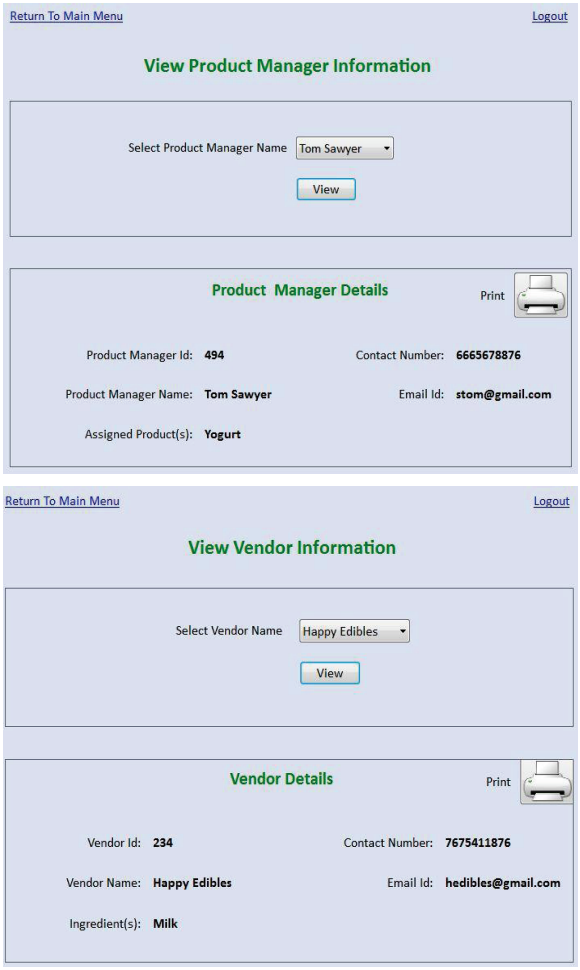

Fig. 5. Virtual windows for product manager and vendor information.

4) **Computer** – Test team member who simulates the virtual windows.

In our study we have three roles (i.e. user, facilitator and log-keeper). Since we developed our prototypes electronically we used a laptop to simulate the virtual windows.

We conducted usability testing with ten distinct users individually in an open lab in Fall 2008. In the first iteration, we selected only one user for the usability testing. In the next iteration we conducted usability testing with two users and in the last iteration we performed usability testing with seven users. We incorporated the corrections from the first and second iterations to the produce a new version of the prototypes in the third iteration. We explained the case study to the users prior to the testing to help them gain the domain knowledge. We used the "observations" data collection method to investigate how the users performed the given task. The advantage of observations is that they may provide deep understanding of phenomenon that is studied. We used category 3 observation approach where there is a high awareness of the user being observed and a low degree of interaction by the researcher [10], [11].

The description for six tasks and its function mini-specifications are given in Table I and Table II. The data

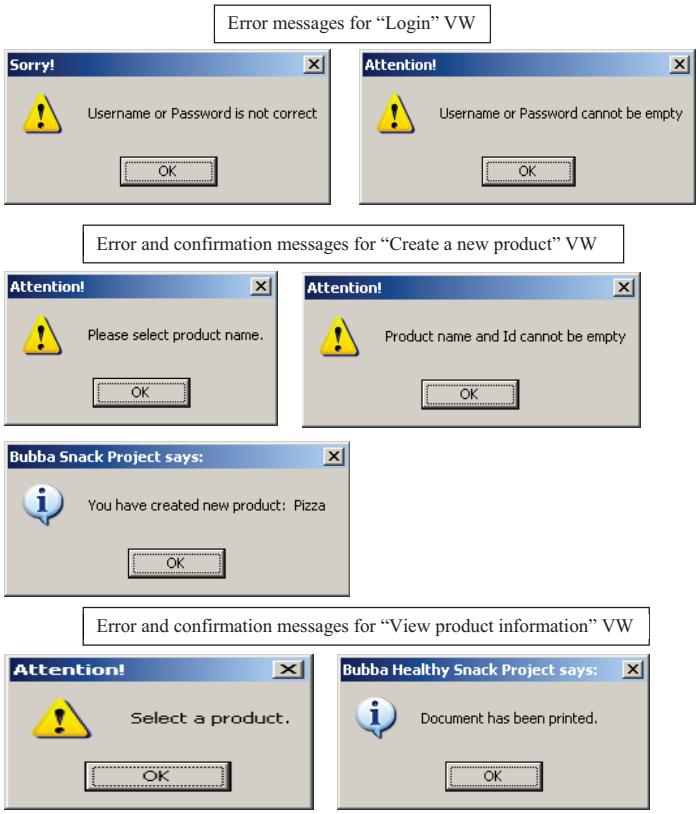

Fig. 6. Confirmation and Error messages (1)

fields presented in the virtual windows were acquired after designing the entity relationship diagram shown in Figure 1. We developed virtual windows in C# using Visual Studio 2008 IDE by following Lausen's gestalt principles [4]. Figures 2, 3, 4, and 5 show the final version of virtual windows of the Bubba's healthy snack project. Figures 6 and 7 show confirmation and error messages. The CREDO in Table III matrix is used to compare the data model and the virtual windows to help find the missing functions is shown in Table III.

Users were given one task at a time. We took the suggestions and opinion after the user finished a given task. The users were requested to "think-aloud" while performing the tasks and this process was video recorded for later analysis. We asked users to give their opinion of the accessing data with prototypes on a Likert scale of 1 to 5.

1 – Not Satisfied
2 – Satisfied
3 – Good
4 – Very Good
5 – Excellent

The log-keeper recorded the number of issues encountered and the time taken to complete a given task using stop watch. Finally, the heuristic evaluation was conducted by an expert following the guidelines of Nielsen [8]. Here are some of the comments.

- Functions or tasks in the main menu may not be clear to the user. There may not be proper affordances for

TABLE IV. TIME TAKEN BY THE USERS TO FINISH TASKS

Tasks	Users (U1–U10)										Mean
	Iteration 1	Iteration 2		Iteration 3							
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	
T1	144	152	149	142	144	149	144	144	140	143	145.1
T2	43	49	44	43	48	48	47	44	44	49	45.9
T3	76	79	72	78	73	73	74	76	72	72	74.5
T4	35	38	38	33	31	33	36	33	35	33	34.5
T5	36	35	34	36	37	38	36	38	38	38	36.6
T6	20	19	22	21	20	22	22	20	21	22	20.9

TABLE V. DESIGN ISSUES

Problem Id	Task Id	Problem description
P1	T1	Not obvious what 'product Id' means.
P2	T1	No option for 'hire a new manager' in the 'assign product manager' dropdown listbox.
P3	T1	Not obvious what is currency of the price? Is it in Dollars or Rupees?
P4	T2	Not clear whether the product is existing product or not.
P5	T2	No option for 'create a new product' in the 'select product name' dropdown listbox.
P6	T2	No email option for product information.
P7	T3	Not clear what 'Manager Id' means.
P8	T3	No 'country' option for the address.
P9	T3	No dropdown list for 'state'.
P10	T3	Not obvious whether the contact number is cell phone/home phone/fax.
P11	T4	Not clear whether the manager is the existing manager or not.
P12	T4	No option for 'hire a new manager' in the 'select product manager name' dropdown listbox.
P13	T4	No email option for product manager information.
P14	T5	No email option.
P15	T6	No print option.

TABLE VI. PROBLEMS ENCOUNTERED BY USERS

Problem	Users (U1–U10)										Hit-rate
	Iteration 1	Iteration 2		Iteration 3							
	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	
P1	Y	Y	Y					Y			4
P2	Y	Y					Y				3
P3		Y									1
P4		Y		Y				Y			3
P5			Y				Y			Y	3
P6		Y				Y	Y				3
P7	Y		Y		Y	Y				Y	5
P8	Y			Y							2
P9	Y										1
P10	Y										1
P11	Y				Y			Y	Y		4
P12	Y					Y					2
P13						Y	Y				2
P14		Y							Y		2
P15		Y							Y		2

the user to interact with the main menu interface.

- Screen size is too small, therefore the visual components of the user interface are cramped, and not pleasant to look at.
- Text input field size may not be sufficient for the user to input the required data.
- A graphic or an icon could be more suited for functions like Print or Send Email along with a caption stating its purpose.
- There is no feedback when Print or Send Email is clicked.
- The names of the tasks may be too ambiguous.
- When displaying information, the captions and the data look too similar. A way of differentiating data and captions is needed.
- Error messages need to be friendlier in tone.
- Error messages need some graphic or symbol to indicate the seriousness of the error, whether it is an invalid operation, or just a confirmation message.
- Error messages have to help the user find out their mistake, by giving more feedback.

IV. RESULTS AND DISCUSSION

To provide the overview of all the collected data during the usability testing, we presented the data in different tabular forms. Table IV shows the time taken in seconds by different users to finish the given tasks. The last column in this table shows the average time taken to complete a given task. Table V shows the description of different design problems associated with different tasks. Table VI presents the relation between design problems encountered by users. The character "Y"

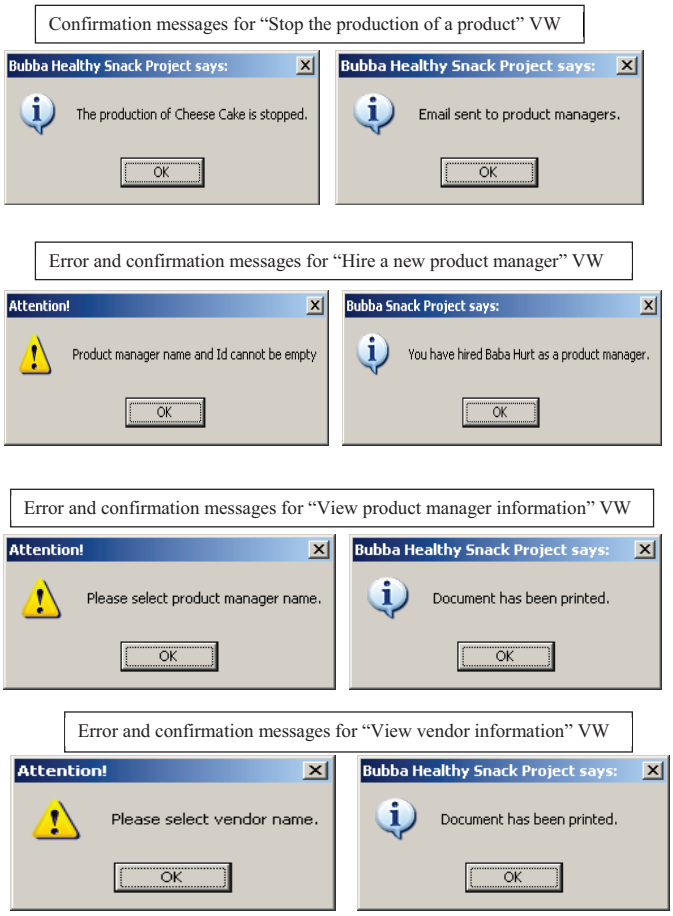

Fig. 7. Confirmation and Error messages (2)

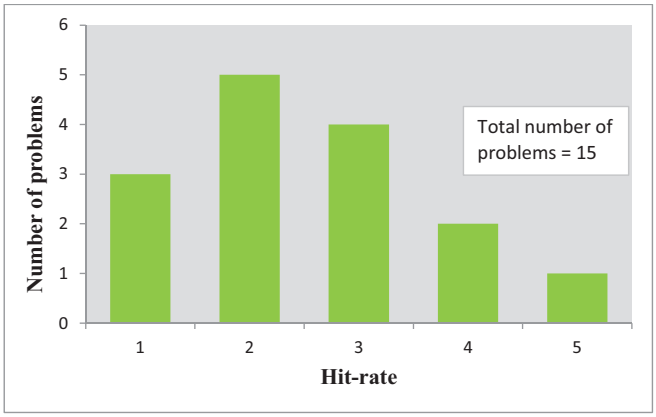

Fig. 8. Number of problems and hit-rates

indicates that a particular problem in the row was encountered by the user in the column. The last column in this table is the hit-rate of a problem. The number of times a problem was encountered by different users is called hit-rate (e.g., problem P1 had a hit-rate of four which means four users encountered the problem P1). The bar graph in Figure 8 provides a concise display of a distribution of number of design problems and the hit-rate. From Figure 8 there is one problem with a hit-rate of 5. Similarly, two problems with a hit-rate of 4, four problems with a hit-rate of 3, five problems with a hit-rate of 2, and three problems with a hit-rate of 1.

This paragraph answers the research questions and highlights our observations.

1) No user finished all the tasks with no un-answered questions. Users finished simple tasks without any questions. Most of their questions related to design issues in complex tasks (T1 and T3). In such cases, the average opinion of the users regarding accessing of data is less than 3 on a Likert scale. Otherwise their opinion is 5.
2) We listed several design problems in Table V identified during usability testing.

Most of the users were expecting the rich functionality of the system even though it is not required for usability testing. Some users identified problems of not having an email option and a print option in most of the prototypes. From Table VI we observe that the number of problems identified in the Iteration 3 are less than the problems identified in the first two iterations.

A. *Lessons learned and Recommendations*

This section explains the different lessons learned in conducting usability testing. This section also highlights the recommendations to software engineering practitioners.

Lessons Learned:

- Communication between the test team members and the users should be limited. This helps the team to observe and concentrate on the user's activities.
- Distinguish between the simple and complex tasks for the users. This allows the user to take sufficient time to complete the given task.
- Developing prototypes and testing them with appropriate users becomes significantly less expensive in developing software.
- Usability testing increases the understandability and re-usability of artifacts. Thus, maintenance of software is relatively easy.

Recommendations:

- Incorporate systematic usability testing where ever applicable in the software life cycle development (e.g., requirements gathering and design user interfaces.)
- Do not select inappropriate subjects (users) for the usability testing. This may distort the results.
- Conduct usability testing on an iterative basis. Incorporate corrections from one iteration to the next iteration.

V. Threats to Validity

Internal validity: Threats to internal validity are influences that can affect the outcome by the selection of the human subjects [11]. The important factors in selecting human subjects for usability testing are the domain knowledge and IT

background. We controlled this threat by explaining the case study to users to give them the necessary domain knowledge.

External validity: Threats to external validity are the conditions that limit our ability to generalize the results outside the scope of our study. There are three types of interactions with the treatment: people, place and time [11]. The interaction of selection and treatment of subjects for our study may not generalize our results to industry applications because our subjects are not from the same domain as our case. To control the interaction of setting and treatment threat, we conducted usability testing in our lab. To control the interaction of history and treatment threat, we did not conduct the usability testing right before and right after the classes of the users. We took appointments from the users and conducted the experiment at those times.

VI. Conclusions and Future work

We illustrated the usability testing from the software engineering perspective using a case study. We also presented several artifacts used in the usability testing. Our results indicate that no users can finish complex tasks (T1 and T3) with no un-answered questions. A tabular view of design issues are provided. These issues help in designing user interfaces. We also noticed that design issues are reduced from first iteration to third iteration. In addition, we highlighted our lessons learned and recommendations for software engineering practitioners using usability testing effectively in other software projects. Our results provide additional validation to the usability testing approach. We plan to extend this study with more complex tasks focusing on different domains, and with a larger number of users on large case study.

Acknowledgment

We would like to thank Rajiv Chowdary Govapudi and David Thomas Samson for their help in preparing virtual windows and for their facilitator and log-keeper roles during the usability testing respectively. A special thanks to Dr. Edward B. Allen for reviewing this paper and for other suggestions.

References

[1] W. F. W. Ahmad, S. Sulaiman, and F. S. Johari, "Usability management system (usemate): A web-based automated system for managing usability testing systematically," in *Proceedings: 2010 International Conference on User Science Engineering*, 2010.

[2] M. Alshamari and P. Mayhew, "Task design: It's impact on usability testing," in *Proceedings: The Third International Conference on Internet and Web Applications and Services*, 2008, pp. 584–589.

[3] K. Beck and et al., *Manifesto for Agile Software Development*, 2001 (accessed May 26, 2013). [Online]. Available: http://agilemanifesto.org/

[4] S. Lauesen, *User Interface Design: A Software Engineering Perspective*. Harlow, England: Pearson Education Limited, 2005.

[5] A. Lodhi, "Usability heuristics as an assessment parameter for performing usability testing," in *Proceedings: 2nd International Conference on Software Technology and Engineering*, 2010.

[6] G. Meszaros and J. Aston, "Adding usability tetsing to an agile project," in *Proceedings: AGILE 2006 Conference*, 2006.

[7] J. Miller, "Usability testing a journey, not a destination," *IEEE Internet Computing*, pp. 80–87, 2006.

[8] J. Nielsen, *10 Usability Heuristics for User Interface Design*, 2005 (accessed May 26, 2013). [Online]. Available: http://www.nngroup.com/articles/ten-usability-heuristics/

[9] P. Runeson and M. Host, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, pp. 131–164, 2009.

[10] F. Shull, J. Singer, and D. I. K. Sjøberg, *Guide to Advanced Empirical Software Engineering*. Berlin: Springer, 2010.

[11] C. Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, and A. Wesslen, *Experimentation in Software Engineering*. Berlin: Springer, 2012.