Structure of a Formal User Model for Construction Information Retrieval

Haiyan Xie, Raja R.A. Issa and William O'Brien, University of Florida, Gainesville, FL 32611-5703,USA
(raymond-issa@ufl.edu)

Summary

Information science researchers and developers have spent many years addressing the problem of retrieving the exact information needed and using it for analysis purposes. In information-seeking dialogues, the user, i.e. construction project manager or supplier, often asks questions about specific aspects of the tasks they want to perform. But most of the time it is difficult for the software systems to unambiguously understand their overall intentions. The existence of information tunnels (Tannenbaum 2002) aggravates this phenomenon.

This study includes a detailed case study of the material management process in the construction industry. Based on this case study, the structure of a formal user model for information retrieval in construction management is proposed. This prototype user model will be incorporated into the system design for construction information management and retrieval. This information retrieval system is a user-centered product based on the development of a user configurable visitor mechanism for managing and retrieving project information without worrying too much about the underlying data structure of the database system. An executable UML model combined with OODB is used to reduce the ambiguity in the user's intentions and to achieve user satisfaction.

1 Introduction

Information science researchers and developers have spent many years addressing the problem of retrieving the exact information needed and using it for analysis purposes. In construction information management context, the users, i.e. construction project managers, superintendents, or material vendors, often ask questions about specific aspects of the tasks they want to perform through some information-seeking dialogues. But most of the time it is difficult for the software systems to unambiguously understand their overall intentions. With the object of design an effective and efficient information retrieval system, we need to study the users' needs under that subject. The users of a construction information system need to manage the vast amount of data and information documentation related to the construction process. The knowledge embedded in the design of complex engineered equipments and systems are essential for project performance. But information loss or data chaos may also happen when there are lots of changes during the design / construction process of a project. If change logs are not kept up to date reflecting all the revisions, contractors risk losing time, money and quality on the project. In addition to having an integrated system storing all the data and information about construction projects, users will also need to have a system to retrieve the requested information.

A lot of information retrieval techniques, particularly those in DBMS systems, typically require users to have enough understanding on software system design; or else, users will have to follow the built-in search and reporting functions the system provides. There should be some adequate user training to let users get used to new systems. But the clumsy searching process of a construction information system may cause users to lose patience. In addition, excessive requirements in terms of the users' ability to use database query languages will affect the users' evaluation of the systems. The existence of information tunnels (Tannenbaum 2002) aggravates

the phenomenon because users may have to use multiple information systems or processes to retrieve some useful information. According to Tannenbaum (2002), information tunnels are typically based on processing requirements. Even though the same data may be required in more than one tunnel, it is usually not easy to go from one tunnel to another. In our case, users have to be able to search for tunnel-resident information from different databases.

The goal of this study is to help project participants in creating a more suitable tool to discover the intended information. To address this problem for the construction domain, the structure of a formal user model for information retrieval in construction management needs to be developed. The information retrieval system explored in this study is a user-centered product based on a user configurable visitor mechanism for managing and retrieving project information without worrying too much about the underlying data structure of the database system.

2 Literature Review

One of the approaches that software developers hope will help people out of the information retrieval confusion is through the design of a user-friendly interface. To reach this goal, several approaches were suggested and some have even been implemented with some success (Johnson 2000; Rudisill et al. 1996). Software Agent technology is one of these approaches. By Microsoft's definition, Software Agent technology is a set of software services that can easily enhance the user interface of applications and Web pages with interactive personalities in the form of animated characters (Microsoft 2003). For example, when a MS Word user types in "Attn:" in a letter, the software agent running in the background reasons automatically that the user wants to insert an address next and it pops up the address searching interface for the user. According to Wooldridge and Jennings (1995), an agent should be able to perceive the environment (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it. Although developers are enriching the abilities of Software Agents, such as by adding Artificial Intelligence (AI) to them, Software Agents still cannot replace human decisions. In addition, the design of the Software Agents is based on modeling the users' behaviors and needs.

A fundamental objective of human-computer interaction research is to provide users with experiences fitting their specific background knowledge and objectives. It is called customized software design. The challenge in an information-rich world is not only to make information available to people at any time, at any place, and in any form, but specifically to say the right thing at the right time in the right way. User Modeling is a technology that addresses the need to personalize information services in a computer system. As Kobsa (2000) pointed out, research on user modeling is usually traced back to the works of Allen, Cohen and Perrault, e.g. (Allen 1979; Cohen and Perrault 1979; Perrault et al. 1978) and Elaine Rich (Rich 1979a; Rich 1979b). Ten years after this seminal research, numerous application systems for user modeling were developed. These models collected different types of information about their current users and exhibited different kinds of adaptation to those users.

Kobsa (2000) has divided user modeling research into three stages. The earliest stage occurred before the mid-1980s, and user modeling during that stage was performed by application systems. No clear distinction could be made between the system components that served user modeling purposes and those that performed other tasks. From the mid-1980s to mid-1990s, such a separation was increasingly made, e.g. (Sleeman 1985), but no efforts have been reported on rendering the user modeling component reusable for the development of future user-adaptive systems. Past work in the area lacks a robustly defined and generally applicable system design that will enable end-users to access personalized information. Since mid-1990s till now, to

address the issues of reusable components and robustly defined system design, current user modeling research has focused on user-adaptive systems. User modeling, software reuse, and organizational memories and organizational learning (e.g., creating socio-technical environments that help to transcend the limitation of the individual human mind) are some of the approaches that combine computer techniques with natural and/or social issues.

Many researches have been done on how to do user research and user modeling. Fischer (2000) addressed the difficulties in user modeling and mentioned the following three categories of user modeling research with high-functionality and usable.

1. Exploring different domains such as: natural language dialog, human computer interaction, intelligent assistants, information retrieval, and high-functionality applications;
2. Identifying important distinctions such as: adaptive versus adaptable components, explicit versus implicit modeling techniques, user models versus task models, canonical versus individual models, and long-term versus short-term models;
3. Creating a number of challenging research problems, such as how to:
    - Integrate different modeling techniques
    - Capture the larger (often unarticulated) context and what users are doing especially beyond the direct interaction with the computer system)
    - Identify user goals from low-level interactions
    - Reduce information overload by making information relevant to the task at hand
    - Support differential descriptions by relating new information to known information and concepts; and
    - Reach a better balance for task distributions between systems and users.

Cooper (2003) discussed user interface design essentials in his book -- "About Face 2.0: The Essentials of Interaction Design". The user interface design belongs to the first category of user modeling – human computer interaction, as grouped by Fischer (2000). Cooper concluded that while technology frees us to perform great feats of invention, it also simultaneously ties us to ways of thinking that are contrary to the natural expression of human behavior. Cooper defined the user model design as: "Understanding the user's wants, needs, motivations, and contexts; Understanding business, technical, and domain requirements and constraints; Translating this knowledge into plans for artifacts (or artifacts themselves) whose form, contents, and behavior is useful, usable, and desirable, as well as economically viable and technically feasible" (Cooper 2003). Information retrieval research has close relationship with user modeling. The general area of information access and retrieval aims at modeling, designing and implementing systems able to provide fast and effective content-based access to a large amount of information. The purpose of Information Retrieval (IR) is to find the information according to requestors' needs and user modeling servers to find out the right ways to response to these needs (Fischer 1993).

When a system plays the role of a consultant in information-seeking dialogues, the extent to which it provides cooperative responses is directly related to its ability to understand the user's plans and goals (Allen 1983). In such kind of dialogues, the determination of the user's intentions is generally quite difficult not only because the inexplicit statement of the questions, but also because some users do not even know exactly what their questions are. In such cases, the user would ask general questions or questions about particular aspects of the tasks. So, in order to reduce the number of hypotheses on their intentions, several sources of information must be exploited, such as domain knowledge; knowledge about the user; and knowledge extracted from the preceding parts of the dialogue. For all the domain knowledge, user

knowledge, and dialogue knowledge collected either from surveys or other methods, there must be some tools to manage the knowledge and use the knowledge to help the system in making default choices among the possible hypotheses on the user's intentions. The User Model (UM) is one such tool that can extract general information about the user from prototypical information, and with some user model acquisition rules, UM can be enhanced by inferring new intentions from the system's hypotheses on the user's plans as well. For example, UM can be enhanced from inferring the user's beliefs and goals from the sentences s/he utters (e.g. through the recognition of presuppositions in the user's utterances) and from her/his reactions to the answers from the system (e.g. if the system uses a certain word and the user does not ask for clarification, the system infers that s/he understands its meaning (Chin 1989)).

3 Problem Statement

The goal of this study is to improve the level of satisfactory results obtained from a conventional information retrieval (IR) system by modeling the user's information retrieval approach and leveraging the user model into structured knowledge. We analyze the users' needs and model them by the "use case" modeling method (Mellor and Balcer 2002), and then exploit the relationships in the use case to address the IR system with a more relevant request. The Use Case model is about capturing user requirements and describing what the proposed system will do at a high-level and with a user focus for the purpose of scoping the project and giving the application some structure. The Use Cases are the smallest unit of delivery. Each increment that is planned and delivered is described in terms of the Use Cases that will be delivered in that increment. Use Cases are not intended to capture all of the system requirements. Use Cases do not capture how the system will do anything - nor do they capture anything the actor does that does not involve the system. For example, in the Construction Material Purchase Order process, when a project manager calls a vendor for some materials' prices, this activity will not be included in the system.

It is notable that in restricted search domains the original query from the user may omit terms that precisely describe the sought topic; only to describe it partially; or the user may lack the vocabulary needed to search a narrow domain effectively. Therefore, this study will concentrate on the user modeling approach to appropriately reflect the users' needs, thereby retrieving more relevant documents. The methods used and their results are described below.

It is not hard for the system developer to retrieve the needed information successfully, but for the ordinary user to figure out how to retrieve the information or even to learn the advanced usage of information management tools is not easy. In this study we proposed to conduct a detailed case study of the material management process in a construction company. Based on the case study, a formal user model will be to reflect the material management process. The proposed user model in this research is based on the implementation of user research in software development, to be specific, the Unified Modeling Language (UML) can be implemented to build up a formal user model to generate a fast and smooth transient from user needs to the software products. Further more, the object-oriented languages, such as Java, can help to build up a configurable visitor pattern to retrieve information. As a design consideration, user metadata is helpful in generating information on demand for Database Management System (DBMS) implementation. The proposed approach will help the user in retrieving information according to their needs, and in reducing the ambiguity in their intentions.

For many construction companies, material purchase is a daily task. The purchaser will need to retrieve information about the price of the material, its inventory information and any other information needed to make decisions and write up a final purchase order. This study proposes an information retrieval UM for material management. Through the analysis of the material management Use Case, a User Model with embedded domain knowledge will be built. It is

proposed that this UM be incorporated into a Data Base Management System (DBMS) to achieve better performance and user satisfaction.

4 Material Management Use Case

The first step in our research is to build up the use case of a specific area. The use case diagram is a model of the system's intended functions and its environment that supports the business process. This model serves as a contract between the customer and the developer. As mentioned before, the material management process was chosen for the use case example because it happens with high frequency in a lot of construction companies. The main scenario for a material management use case is as shown in Table 1. The Project Manager is responsible for checking the storage of materials, placing the purchase order, and the maintenance of the material record database.

Table 1 reflects the Project Manager's view of the material management process. A more detailed model of the whole purchasing process is shown in Table 2. A detailed explanation of the functions involved in Table 2 is shown in Table 3. Although the use case presented in Tables 1-3 is not a full-fledged material management business scenario, the idea behind doing this scenario is to start with the modeling of the users doing the information retrieval task and then incorporating the user model into software development.

Table 1 Material management business use case

Project Manager Doing Material Handling Order
1. Project Manager requests a new Material Handling Order (MHO) number. 2. The system prepares a blank MHO form and pulls in a list of available materials in the warehouse from the Material Catalog System. 3. Project Manager selects primary and alternate materials. 4. For each material, the system verifies that the Project Manager has the necessary authorization and adds the material to the MHO form, marking the material with the project job number. 5. When the Project Manager indicates the MHO form is complete, the system saves the form.
Extensions: (The first number indicates the corresponding item above) 1a. Material Catalog System does not respond. The system notifies the Project Manager and terminates the use case. 1b. Project Manager already has a MHO form: System brings up the current version of the Project Manager's MHO form for editing instead of creating a new one. 4a. That material is not enough for the quantity the Project Manager specified. System notifies the Project Manager and asks whether to continue or exit. 4a-1. If the Project Manager wants to continue System adds the material to the schedule using the existing amount 4a-2. If the Project Manager does not want to continue System disables selection of that material and notifies the Project Manager.

Table 2 Formal Use Case for modeling material management purchasing process users

Purchase Material or Products
Primary Actor: Project Manager
Goal in Context: Project Manager buys material through the system, gets it. Does not include paying for it.
Scope: Business – The overall purchasing mechanism, electronic and non-electronic, as seen by

the people in the company.
Level: Summary
Stakeholders and Interests Project Manager: Wants what he/she ordered and an easy way to do that. Senior Project Manager: Wants to control cost but allow needed purchases. Vendor: Wants to get paid for any goods delivered.
Precondition: None
Minimal Guarantees: Every order sent out has been approved by a valid authorizer – Senior Project Manager or President of the company.
Trigger: Project Manager decides to buy something because s/he finds the requirements of the materials on the drawings or gets the requests from project superintendent for the materials.
Main Success Scenario:
1. Superintendent / Project Manager (Requestor): initiate a request.
2. Senior Project Manager (Approver): Check money in the budget, check price of products, complete request for submission.
3. Project Manager (Buyer): Check contents of storage, find best vendor for products. Validate Approver's signature. Complete request for ordering. Initiate Purchase Order with Vendor.
4. Vendor: Deliver products to jobsite. Get receipt for delivery (Most of the times, Vendor prepared the receipt and requests the jobsite superintendent to sign it. This is out of scope of system under design).
5. Jobsite superintendent (Receiver): Sign the receipt. Send a copy of receipt to Project Manager.
6. Superintendent / Project Manager (Requestor): Mark request delivered.
Extensions
• 1a. Requestor does not know vendor or price: leave those parts blank and continue
• 1b. Prior to receiving goods, Requestor can change or cancel the request. Canceling it removes it from any active processing. (Delete from the system) Reducing price leaves it intact in process. Raising price sends it back to Approver.
• 2a. Senior Project Manager does not know vendor or price: Leave blank and let Project Manager filling in or call back.
• 2b. Senior Project Manager is not Requestor's manager: Still okay, as long as Senior Project Manager approves.
• 2c. Approver declines: Send back to Requestor for change or deletion.
• 3a. Project Manager finds goods in storage: Send those up, reduce request by that amount and carry on.
• 3b. Project Manager fills in Vendor and price, which were missing: Request gets re-sent to Senior Project Manager.
• 3c. Request involves multiple vendors: Project Manager generates multiple POs.
• 3d. Project Manager merges multiple requests: Same process, but mark PO with the requests being merged.
• 4a. Vendor does not deliver on time: System does alert of non-delivery.
• 5a. Partial delivery: Receiver marks partial delivery on PO and continues.
• 5b. Partial delivery of multiple-request PO: Receiver assigns quantities to requests and continues.
• 5c. Products are incorrect or improper quality: Requestor refuses delivered goods.
Technology and Data Variations List: None
Priority: Various
Release: Several
Response Time: Various
Channel to Primary Actor: Internet browser, mail system, or equivalent
Channels to Secondary Actors: Fax, phone, paper proposal

Open Issues:
When is a canceled request deleted from the system?
What authorization is needed to cancel a request?
Who can alter a request's contents?

Table 3 Description of some of the related functions in the use case

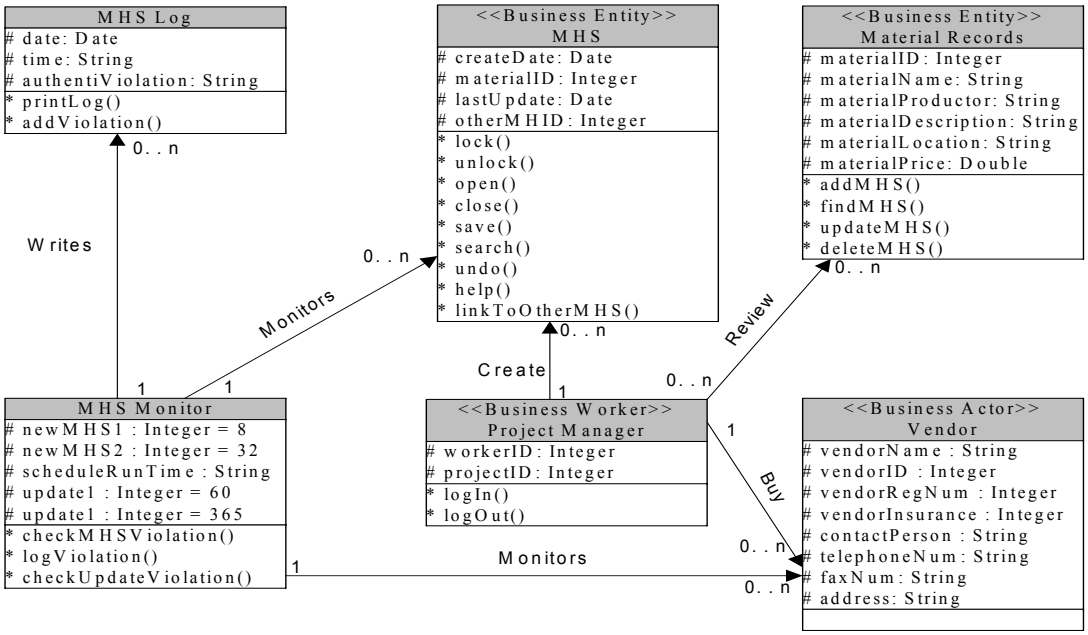
Actor	**Description**
Project Manager	Person monitoring and controlling construction project
Use Case	**Description**
Login	Log Project Manager into system
Logout	Log Project Manager out of system
Set Monitor Parameters	Allow Project Manager to specify boundaries and precision of items being monitored
Display Results	Display the results that the user selected according to the specified preferences.

5 UML Implementation Of Information Retrieval User Model For Material Management

The introduction of new technologies has led to a range of customized products with complex and embedded functionality. Increasingly products are being designed to handle a number of complex tasks. Such products incorporate the functionalities with which the user can interact only indirectly through an interface. As a consequence, users often struggle with the complexity of the interface instead of interacting with the content, which is what the user is actually most interested in.

The goal of designing user-centered products is to establish ways in which a product may increase the ease with which a user communicates with that product. The design concepts and prototypes should demonstrate the overall user experience and quality interaction. User-product communication design concepts should be as far reaching as possible towards exploring new paradigms of user-product interaction. What is proposed in the rest of the paper is to incorporate the information retrieval user model into the software system design. After satisfying the information needs of the users as defined in the user model and building up the prototype of a system, developers should go back to the expected users and listen to their opinions, and then make changes to the original user model and re-evaluate it again. In the material handling use case, we have already analyzed the PM's information needs and the different relationships involved in the Material Handling System (MHS). Figure 1 translates the use case in Tables 1-3 into a typical software developer's view. For the example described in Tables 1-3 and Figure 1, the possible Java class for MHS is as shown in Figure 2.

Figure 3 shows the system design process. In our illustrated use case, the material management process, we have finished Use Case Analysis, built up a formal user model for the use case, and derived the software functionality and specifications. The User Model between the Graphical User Interface (GUI) and Code shown in Figure 3 has different meaning from the formal user model (which was used to build up the Code). It stores all the user preferences. When the user logs in, it will know which user class this user belongs to, i.e. project manager or superintendent. Then it will choose the presentation format of the system operation result for the user. To fit the various requests from users, we propose to use a configurable visitor pattern. After the formal user model for material handling in a construction company is built, the next step is to proceed with this user-centered design to link the functions of the Java class (shown in Figure 2, for example) with the DBMS. At this point the use of a configurable visitor design pattern is proposed.



Note: # -- means the item after it is a parameter of the object.
* -- means the item after it is a function / method of the object.
1 / (0..n) --> 1/ (0..n) – means the relationship is 1 to 1, 1 to many, many to 1, or many to many.

Figure 1 Formal Class Diagram for Detailed Use Case Analysis

```
public class MaterialHandling extends StateMachine {
        private StateMachineState currentState;
        private Date createDate;
        private Integer materialID;
        private Date lastUpdate;
        private Integer otherMHID;
        .
        .
        protected void lock() throws Exception;
        protected void unlock() throws Exception;
        . . .
        }
```

Figure 2 Java Class Created by Applying Tables 1-3 and Figure 1

Grand (2002) discussed the visitor pattern and the possible ways to implement this pattern in his book--"Patterns in Java." A visitor pattern allows the client object to be closely coupled to the construction of the new complex object. Instead of describing the content of the objects to be built through a series of method calls, the information is presented in bulk as a complex data structure. A visitor pattern can be used to encapsulate operations in a single class that would otherwise be spread across multiple classes. It also allows encapsulating the logic for simple manipulations of a parse tree in a single class. Grand mentioned, "One way to implement an operation that involves the objects in a complex structure is to provide logic in each of their classes to support the operation. The Visitor pattern provides an alternate way to implement such operations that avoids complicating the classes of the objects in the structure by putting all the necessary logic in a separate class. The Visitor pattern also allows the logic to be varied by using different Visitor classes" (Grand 2002).

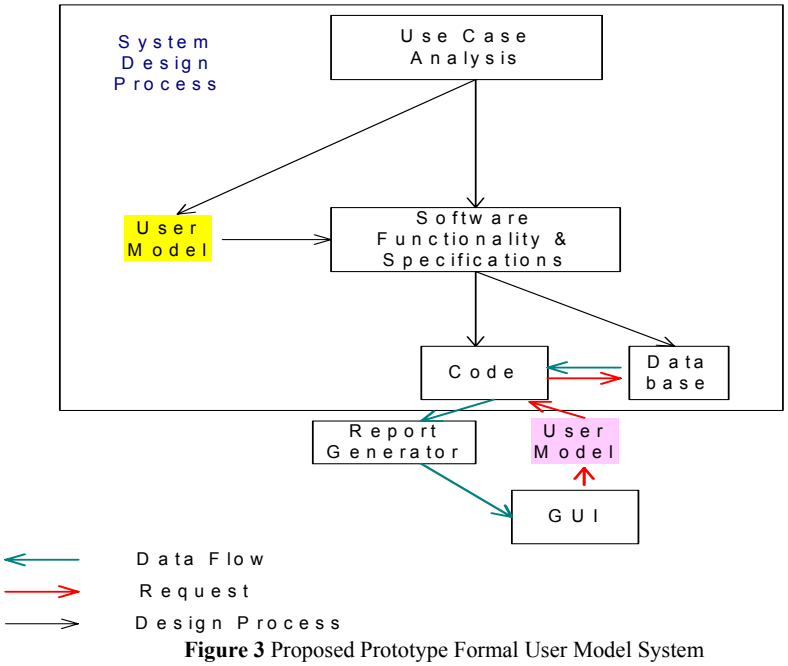


Figure 3 Proposed Prototype Formal User Model System

In the proposed system, implementation of the following visitor classes is planned:

1 **InformationSearcher.** This class starts the entire visitor process and the information retrieval task.

2 **ProjectVisitor.** This is an abstract class and provides logic for its subclasses. Its subclasses explore the objects that constitute the entire data structure, a construction project tree, on which we have organized all the information regarding a construction project. The concept is that instances of subclasses of the `ProjectVisitor` class visit the objects and gather information from each object, and then operate on the information.

3 **TOCVisitor.** This is a subclass of the `ProjectVisitor` class. It is responsible for generating a table of contents. It works by examining each `Division` object directly owned by a ConstructionProject object. When it finds a Division object with a style that is in the internal TOC table, it generates a corresponding TOC entry. The TOCVisitor may go very deep down to the leaf node of the tree data structure. From a parent node, when TOCVisitor finds the child node object match with the searching condition, it will use the child node to replace the existing TOC entry.

4 **ReportVisitor.** This subclass of the `ProjectVisitor` class is responsible for reporting back to ReportManager process. It begins by being told to look for sub-nodes that correspond to certain levels of organization in a ConstructionProject object. It finds those levels of organization in the internal TOC table. It fetches the styles associated with those levels of organization from the table. It then follows the pointers and looks for objects that have the styles it fetched from the table. When it finds an object with one of those styles, it creates a new ConstructionProject object. It writes the new ConstructionProject object and all the objects associated with it to a file. Besides the pointers to the project data tree, the TOC table also stores the pointers to Inlist and Outlist. When the ReportVisitor writes the new ConstructionProject object and the associated objects into a file, it will store the pointer address in the TOC table. After all the requested new ConstructionProject objects are stored into their specific files and the addresses of the files are saved in the TOC table, the ReportVisitor will call the ReportManager to let the ReportManager access the TOC table and retrieve the files. Subsequently, when new addresses replace the old addresses in the

Inlist column of the TOC table, the “kicked-out” addresses will be saved in the “pointers to the Outlist” column. Once the capability limitation of the TOC table is reached, the first saved address in the “pointer to Outlist” column will be deleted.

6 Conclusions And Future Work

User model acquisition is a difficult problem. In Machine Learning, the information available to a user modeling system is often limited, and it is hard to infer assumptions about the user that are strong enough to justify non-trivial conclusions. In Information Retrieval, user models have been limited to lists of terms relevant to an information need. In the proposed research, the user model will be built based on interviews with prospective users, case studies found in literature and personal industry experience. Through the analysis of the use case of material handling process, the feasibility of involving users in the life cycle of software system development will be explored. The final Java classes are built up based on the use case → user model → Java code evolvement. The users could evaluate the temporary result system and according based on their feedback, the use case and user model could be changed and so would the software system. The next step in this research will include doing user evaluation of the prototype system and adjusting the system based on their comments. In the future, research on the executable user model with semantics for actions will also be explored.

7 References

Allen, J. (1983). Recognizing intentions from natural language utterances, MIT Press.

Allen, J. F. (1979). A Plan-based approach to speech act recognition. 131/79, Dept. of Computer Science, University of Toronto, Canada.

Chin, D. (1989). KNOME: Modeling what the user knows in UC, Springer Verlag, Berlin.

Cohen, P. R., and Perrault, C. R. (1979). Elements of a Plan-based theory of speech acts. Cognitive Science, 3, 177-212.

Cooper, A. (2003). About Face 2.0: The Essentials of Interaction Design, Wiley Publishing, Inc., Indianapolis, IN.

Fischer, G. (1993). Shared Knowledge in Cooperative Problem-Solving Systems - Integrating Adaptive and Adaptable Components, Elsevier Science Publishers, Amsterdam.

Fischer, G. (2000). User Modeling in Human-Computer Interaction. UMUAI 10th Anniversary Issue.

Grand, M. (2002). Patterns in Java: a catalog of reusable design patterns illustrated with UML, Wiley Pub., Indianapolis, Ind.

Johnson, J. (2000). GUI Bloopers: Don'ts and Do's for Software Developers and Web Designers, Academic Press, San Diego, CA.

Kobsa, A. (2000). Generic User Modeling Systems.User Modeling and User-Adapted Interaction 11, 49-63.

Mellor, S. J., and Balcer, M. J. (2002). Executable UML: A Foundation for Model-Driven Architecture, Addison-Wesley, Boston.

Microsoft. (2003). <http://www.microsoft.com/msagent/default.asp> (April 16 2004).

Perrault, C. R., Allen, J. F., and Cohen, P. R. (1978). Speech acts as a basis for understanding dialogue coherence. 78-5, Department of Computer Science, University of Toronto, Canada.

Rich, E. (1979a). Building and Exploiting User Models, Ph.D. Thesis, Carnegie-Mellon University, Pittsburgh, PA.

Rich, E. (1979b). User modeling via stereotypes. Cognitive Science, 3, 329-354.

Rudisill, M., Lewis, C., Polson, P., and McKay, T. (1996). Human-Computer Interface Design : Success Cases, Emerging Methods and Real-World Context, Morgan Kaufmann Publishers, Inc., San Francisco, CA.

Sleeman, D. (1985). UMFE: A user modeling front-end subsystem. International Journal of Man-Machine Studies, 23, 71-88.

Tannenbaum, A. (2002). Metadata Solutions: Using Metamodels, Repositories, XML, and Enterprise Portals to Generate Information on Demand, Addison-Wesley, Boston.

Wooldridge, M., and Jennings, N. R. (1995). Agent Theories, Architectures, and Languages: a Survey, Springer-Verlag, Berlin.