Supporting Workflow in User Interface Description Languages

Nicole Stavness
Department of Computer Science
University of Saskatchewan
Saskatoon, Saskatchewan, Canada
+1 306 966 8654
nicole.stavness@usask.ca

Kevin Schneider
Department of Computer Science
University of Saskatchewan
Saskatoon, Saskatchewan, Canada
+1 306 966 4891
kas@cs.usask.ca

ABSTRACT

XML-based user interface description languages (UIDLs) have been developed to support user interface portability across multiple platforms. UIDLs express various aspects of the user interface, including the abstract and concrete elements of the user interface, the tasks to be performed by the user, and the user interface dialogue.

We have developed the progression model for expressing workflow aspects of an interactive system using an XML-based language. The progression model considers workflow to be a sequence of *scenes* progressing towards an organizational goal. The model allows us to express workflow explicitly using a markup language.

In this paper we present a prototype system, the progression analyzer that accepts a progression, renders the user interface described by a *scene* and provides the user with a mechanism to monitor, save, recall, reorder and coordinate the workflow.

Keywords

Workflow, Task Model, User Interface Description Language, XML

INTRODUCTION

Business organizations use interactive information systems to support their business processes. Users require flexibility when interacting with the system to contend with changes in the business processes, to support differing work approaches, and to coordinate the activities of various workers. This flexibility can be supported by workflow systems.

UIDLs specify important aspects of the user interface. Unfortunately, UIDLs do not express aspects of the user interface related to workflow. We have developed the progression model [18] to explicitly express workflow as a sequence of *scenes* called a *progression*. Our language to express progressions is XML-based.

Our approach is specifically aimed at the development of transaction-based interactive systems. A progression in this context may be renting a car, filling out a mortgage application, or booking a flight.

In this paper we present a prototype system, the progression analyzer, for rendering and interacting with progressions. After a discussion of the related work and introducing the progression model we describe the progression analyzer prototype. We then evaluate our approach by showing how it maps to the components commonly found in a workflow system, using examples from our prototype. We conclude the paper with a discussion of future research directions.

RELATED WORK

In this section we describe workflow models, task models and how task and workflow models differ with respect to user interface development. As well we briefly describe some current XML-compliant UIDLs.

Workflow Model

The workflow model is used to represent the flow of work within a department, across a company or to external agents [14]. The Workflow Management Coalition (WFMC) defines workflow as the automation of a business process, in whole or part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules [20]. The details of a specific business process are defined in a process definition. This includes the sequences of activities and associated relationships; start and finish criteria; and information such as executers of manual or automated activities, procedural rules, and control data. The process definition may also include sub-process descriptions.

Workflow research focuses on approaches to making changes during the business process. Procedure-like, routine processes that are statically supported are on one end of a continuum, and highly unspecified, dynamic processes are on the other end [5]. In adaptive workflow

management systems [13] the procedural rules can be changed or created during the process. Some research proposes a cooperative hypermedia system, with process support through a meta-model, to integrate the efforts towards communication, coordination, and cooperation in workflow systems [7]. A two-part classification defines types of possible flexibilities that may be desired in workflow management applications [9]. 'Flexibility by selection' provides the user some leniency in executing a process by offering multiple execution paths. Alternatively, 'flexibility by adaption' provides the ability to add extra execution paths through additional functionality and tools that allow the workflow type to change and integrate during runtime.

Some workflow concepts that are common in many systems have been identified in [12]. This research has focused on applying workflow to object oriented systems; the following concepts are identified as important to workflow.

- *Monitoring* for contributing information about the circumstances of workflows during execution;
- *History* of workflow actions for evaluation or recovery;
- *Persistence* to save the historic information and provide access to it;
- *Manual Intervention* for changing the order that activities are performed in as they are performed;
- *Worklist* to coordinate the activities among the workers;
- *Federated Workflow* addresses the issue of how workflow systems interoperate.

Task Model

Task Models are logical descriptions of activities that are designed to be carried out in reaching user's goals in an interactive system. There are many different approaches to task modeling such as Hierarchical Task Analysis [2], GOMS [6], UAN [8], and ConcurTaskTrees [14].

Hierarchical Task Analysis (HTA) is based on describing the set of goals, tasks and operations in logical structures of different levels. GOMS (goals, operators, methods, selection) depicts procedural knowledge or 'how-to-do-it" knowledge through fine-grained operators that are performed to reach a goal. UAN (User Action Notation) also follows a hierarchical structure. It provides a notation for designers to describe the dynamic behaviour of graphical user interfaces, where the tasks are represented asynchronously with operators that denote the temporal relationships. The ConcurTaskTrees notation was created to support engineering approaches to task modeling. Temporal relationships are also incorporated for enabling, concurrency, disabling, interruption, and optionality. Additionally, synchronized tasks where the output information of one task is the input information of another are supported.

In relation to business processes, task models describe the paths of activities available to reach the user's goals. Unfortunately, task models often result in large specifications with more detail than is necessary for a designer. Recent research has investigated annotating task models with data artifacts to better support information systems and extracting dialog models from the task model to better support automated generation of user interfaces [10].

Workflow models and a task models both describe how to accomplish work or tasks. As identified in [19], they both have similar concepts, such as actions/tasks and workers/users. Alternatively, [19] points out that workflow models are useful for group or organization interaction, while task models focus on individual users. This coincides with our research that associates workflow models with focusing on the management of task accomplishment processes. In groups or organizations more direction is required to ensure that orderliness and goal accomplishment are maintained. Workflow research as discussed in [12] goes beyond the actual task and provides a meta-level that focuses on how to coordinate the activities towards completion and how to examine them thereafter.

User Interface Description Languages

Souchon and Vanderdonckt [17] have analyzed a number of XML-compliant languages for defining user interfaces including UIML [1], AUIML [4], XIML [16], Seescoa XML [11], Teresa XML [15], and WSXL [3].

User Interface Markup Language (UIML) allows the user to specify the user interface in general terms then render it according to a style description. Abstract User Interface Markup Language (AUIML) focuses on describing the desired user interaction in terms of its purpose rather than appearance. The eXtensible Interface Markup Language (XIML) affords the ability to describe a user interface without concern for the implementation. Software Engineering for Embedded Systems using a Component Oriented Approach (Seescoa XML) defines an XML description to express an abstraction of the user interface using Java User Interface components. Teresa XML provides a facility to support the design and generate a concrete user interface for a specific type of platform. Web Services Experience Language (WSXL) focuses on a web services model to interact with web applications. The User Interface Description Languages referenced above do not address workflow issues.

PROGRESSION MODEL

The progression model [18] incorporates workflow features into a markup language specification. This research is not concerned with the actual rendering of the user interface as is addressed in many other UIDLs. The progression model makes explicit the steps and transactions a user makes when using a transaction–based information system. As the user progresses towards accomplishing a task or goal, the

progression model infrastructure records each step and the state of the transaction.

Making the steps and transactions explicit allows the user to group transactions into batches for later processing, to store partial transactions for later editing, and allows the user to browse historical progressions. Linking the steps in the workflow directly to the transaction provides a means to integrate the process model and the data model in one coherent model. This enables the support of the flow of work for an individual user by supporting new interactions. A series of definitions outline the basic aspects of a progression. Consequently, new interactions are enabled to provide flexible business process support.

Definitions

The following definitions describe the key elements of the progression model and how they relate to each other. These items are graphically depicted in Figure 1.

Progression. A *progression*, p, is a sequence of scenes (or steps), s, in a process to create a transaction, that is $p = < s_1, \ldots, s_n >$.

Progression Interval. A *progression interval*, pi, is a subsequence of a progression or a couple steps.

Scene. A *scene*, s, corresponds to a step in a progression. Each scene of a progression is associated with the user interface, u, current state of the transaction, t, and current state of the workflow, w, therefore, $s = < u, t, w, >$. A scene captures the process and associated data as a user performs actions throughout a progression.

User Interface. The *user interface*, u, is a rendering of the user interface for the current scene. The user can perform user actions, a, according to the components, such as a text field or select box, available in the user interface.

User Actions. The *user actions*, a, for a scene are the interactions that the user performs within the user interface.

Transaction. The *transaction*, t, models the accumulation of information at each point in the progression. Each transaction is made up of a series of *elements*, e, that are accumulated throughout the progression by user actions, that is $t = < e_i, \ldots, e_n >$. As the scenes change, the element additions, deletions, and changes are reflected in the transaction. For instance, if the user is filling out a wizard form, at every submission the new information is added to the transaction. A series of zero or more *user actions*, a, can be performed directly on the transaction. For example, the user may want to directly edit a field in the transaction rather than going back and editing through the user interface.

Workflow. The *workflow*, w, is a sequence of scenes progressing towards an organizational goal. It identifies the scenes that are completed, currently in progress, and not yet started. It also defines who is assigned to complete a scene. Additionally, it outlines the available *workflow actions*, wa, for the current scene.

Workflow Action. A *workflow action*, wa, is an action that affects the workflow of the progression. One type of workflow action is "transform", which may send information to the transaction and change the user interface to a new scene. For example, when the user clicks on a submit button, a new scene is generated. The information that was entered in the previous scene, such as the text entered in a form is reflected in the transaction. The feedback is then displayed to the user through the user interface. The other type of workflow action deals with interacting with the progression. For example, the user can recall a past progression, replay a progression, save a progression, and so on.

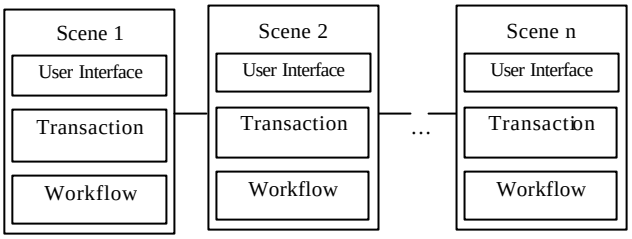

Figure 1. A progression is a sequence of scenes. The rectangles represent scenes that encompass a user interface, a transaction, and workflow.

Benefits of Recording Progressions

An information system is developed to support an organization's business processes. This requires a high degree of flexibility, which has been traditionally difficult to support. The process and data information that is captured through the progression model can be used to support flexible business processes. It is facilitated by displaying the transaction to the user, in addition to the original interface, accompanied by new functionality. Through opening the model to the user in this way, a number of new interactions become available to the user. The interactions that are enabled include: information orientation, immediate updates, historical review, concurrent process comparison, progression batching, and progression manipulation.

Progression Orientation

By visually observing the transaction, the user is able to see the information being built up while the progression is enacted. This provides a reference for the user to ensure the information is correct. Additionally, foresight into the information that is required later in the progression is available from the beginning. This allows users to organize and anticipate the work required to complete the progression. Users that are new to the system now have the ability to reduce the unknown aspects of the system.

Progression Updates

Direct editing of the accumulated data is available while enacting a progression. A user can change information at

any time without having to go backward in the progression and forfeit the later information, such as in web browsing. This also allows the user to keep track of their placement within the progression. Updates may not be allowed for some information items as the constraints of the system must be upheld. Nonetheless, flexibility to make direct changes to the information already accumulated is afforded.

Progression History

A user has access to the progression history. History includes the progression scenes, as well as the transaction snapshots. The user can benefit from the ability to change, replay, or reuse historical information. Changing the history allows the user to move backward in the progression to undo actions. Replaying a progression may be useful for learning how progressions were previously completed by others; remembering what the user did last time they went through the progression; or for the supervisor to look at the work that an employee has performed. New or infrequent users would find the most benefit from this interaction. It is also useful for lengthy progressions, to view work that is not easily remembered. By saving the history of the progression, partial progressions can be closed and returned to at a later time or parts of saved progressions can be reused in future progressions. This is useful when the user is interrupted during a session before they can complete the progression. Alternatively, when the work requirements are more ambiguous and combinations of different progressions are useful.

Multiple Progressions

Multiple progressions can be used to process transactions concurrently. The ability to duplicate and/or view more than one progression at the same time allows for easy comparisons without having to lose work that is already completed, such as when trying out different scenarios or outcomes. The user can go through one possibility, then without losing that information try out another scenario. The outcomes can be considered in a side by side manner.

Progression Batches

Progressions can be applied to multiple items to enable the user to perform a progression and have it affect more than one selected item in the system. For example, a user can perform a progression to change an employee's salary, but have it apply to ten employees. This is beneficial for saving time and consistency while managing large amounts of information.

PROGRESSION ANALYZER

The progression analyzer is a prototype system for displaying information about a progression. A progression is modeled using a mark-up language. Figure 2 shows the outline of the progression model in the markup format. Each scene consists of an abstract user interface (aui) with the corresponding user actions; a set of transactions with possible transaction actions; and finally the workflow with available workflow actions.

The parsed code is interpreted and explicit sections are depicted in the progression analyzer interface. Figure 3 shows a screenshot of the prototype without content.

```
<progression>
    <scene>
        <aui>…</aui>
        <transactions>
            <transaction>…</transaction>
        </transactions>
        <workflow> … </workflow>
    </scene>
    …
</progression>
```

Figure 2. Skeleton of mark-up language for the progression model.

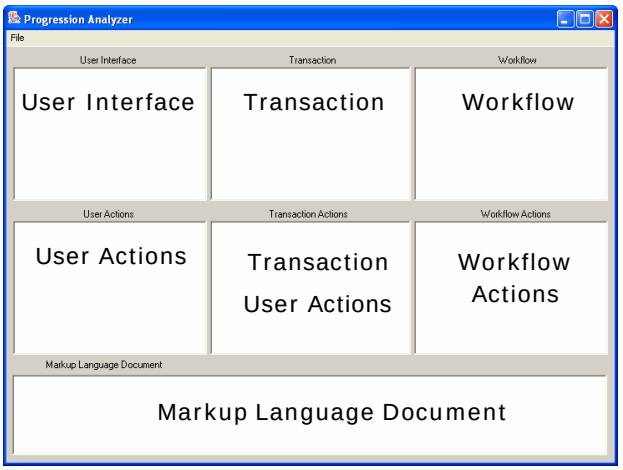

Figure 3. An outline of the progression analyzer panels .

The user interface is rendered in the user interface panel. From the user interface information, the specific user actions are extracted and displayed in the user actions panel. The transaction is shown in a table indicating the transaction number, transaction structure, and the status of the transaction. Any transaction actions that are permissible, such as directly edit field, are presented in the transaction actions panel. The workflow is presented as a table with the scene number, scene name, worker assigned to complete the scene, and status of the workflow scene. Then the possible workflow actions, such as reorder and history, are displayed in the workflow actions panel. Additionally, the mark-up language document is displayed in the corresponding panel.

When the user selects the "transform" workflow action to create and display the next scene, the markup language for that scene is derived from the previous scene and the user actions. The new scene is added to the markup language document and displayed in the progression analyzer panels.

User Interface

The user interface panel shows the rendering of the user interface as it appears to the user. It is non-editable and intended to show the user the snapshot of the user interface at the beginning of the current scene.

User Actions

The user actions panel shows the user all the possible actions that are available to perform within the user interface. The user can interact with the elements on this form and perform user actions toward the progression. For example, the user can enter some text in a text field or select from a select box and so on.

Transaction

The transaction panel depicts the transaction in a table. The transaction consists of all the required information that is accumulated throughout the progression to successfully submit and complete the progression. Each transaction is numbered for unique identification. There is also a status field to indicate the current transaction state, such as partial, complete, valid, or invalid. The remaining fields depend on the information requirements of the transaction.

Transaction Actions

The transaction actions panel displays the available actions that the user can perform on the transaction structure itself. For example, the user may want to edit a field in the transaction directly rather than going through the user interface. System constraints may restrict the user from performing some transaction actions.

Workflow

The workflow panel shows the scenes that are completed during a progression. The workflow panel lays out to the user the scenes that are completed, the scene to be completed next, and the scenes that still need to be completed. Each scene is numbered for unique identification. The status column provides the information on whether the scene is completed, currently being worked on, or yet to be completed. The worker column indicates which human user is assigned to complete the scene actions. This could also be extended to include jobs that the system must complete to show the interaction with the application.

Workflow Actions

The workflow actions panel shows all the possible workflow actions that the user can perform during the interaction with the user interface. Some workflow actions that we have identified as interesting are: move to the previous scene; move to the next scene; reorder the scenes within the limits of system constraints; transform the scene as the user actions are complete; add the transaction to a batch; save the partially completed progression; and view the progression history.

THE PROGRESSION ANALYZER AND WORKFLOW

Manolescu's research [12] maps workflow concepts to an object oriented framework. In his research he identified six components that are common to workflow systems, namely: monitoring, history, persistence, manual intervention, worklist and federated workflow. In this section a comparison is made between each of these workflow components and the progression model using examples from our prototype.

Monitoring refers to gathering information on the state of the workflow regarding the progress of the activities within the workflow. In the progression model, this relates to presenting the user with information on what work has been done, what is currently being done, and what work remains to be done.

In the example depicted in Figure 4 and Figure 5, the first four scenes are completed in the progression and four remain uncompleted. The worker named Jen is responsible for completing the last two scenes of the progression. She wants to see how far along the other workers are in completing the scenes. By viewing the workflow panel, as shown in Figure 4, she can see that the first four scenes are completed and the fifth is in progress.

Workflow

Scene	Name	Worker	Status
1	Name	Tara	Complete
2	Address	Tara	Complete
3	Account	Jim	Complete
4	Billing	Mark	Complete
5	Shipping	Dave	In Progress
6	Approved	Mark	Inactive
7	Credit	Jen	Inactive
8	Confirm	Jen	Inactive

Figure 4. The workflow panel depicting the status of the workflow.

Additionally, when she is completing her scenes, she can view the buildup of information in the transaction. In Figure 5 she has just entered the address information and then selected the transform workflow action. When she looks at the transaction in third scene, she can see the information that resulted from her actions in the previous scene through the transaction panel. Her address information is added to the correct fields in the transaction.

History refers to recording the actions that were taken during the execution of the workflow. This can be used for evaluation and analysis, as well as information recovery. In the progression model, this history is captured in the evolution of the markup language document. The progression analyzer is intended to allow access to the information through the history workflow action.

Figure 5. A transaction near the end of a progression.

For example, the worker named Dave may want to go back to the previous scene to make a change to his user actions. He can select the previous scene workflow action, which processes the markup language for the previous scene. Figure 6 shows the selection of the previous scene workflow action.

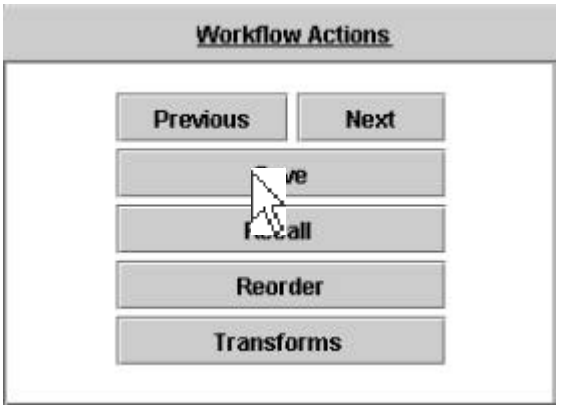

Figure 6. The workflow actions panel, which is displaying the available workflow actions.

Persistence refers to the storing and accessing of the captured history information. This research identifies that persistence and history are often combined in traditional workflow systems. [12] however, provides a separate persistence component, which gives access to the database. In the progression model, the information is saved in the mark-up document. There are also workflow actions that allow partially completed progressions and their transactions to be saved, and then recalled at a later time.

For example, the worker named Mark might want to review what he did in a previous progression. He would have previously selected the "save" workflow action to save the partially completed progression. Then when he wants to re-open the progression he selects the "recall" workflow action and the progression is displayed at the point where he saved it. He can traverse through the progression to replay his actions using the next scene and previous scene workflow actions. Figure 7 shows the file chooser for the recall workflow action.

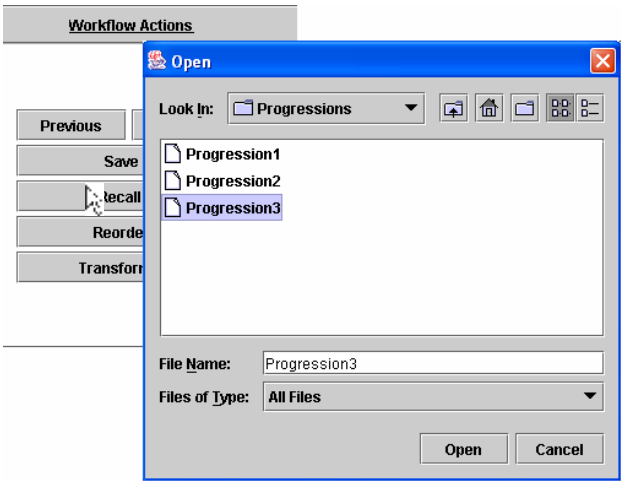

Figure 7. File chooser prompt for the recall button.

Manual intervention refers to allowing users or developers to change the organization of activities during the execution of the workflow. In the progression model, a "reorder" workflow action is provided. It allows the user to perform the workflow scenes in varying orders. This reordering action is limited to the constraints of the system.

For example, the worker named Tara may decide that she does not have the required information to complete scene two – enter credit card information, but she does have her personal information, which is required for scene three. Therefore, she would like to rearrange the scenes so she can complete as much information as possible. She would select the "reorder" workflow action to perform the third scene before the second. Figure 8 shows the new workflow panel resulting from Tara selecting the "reorder" workflow action and switching the scenes.

Figure 8. The rearranged workflow scenes.

Worklist refers to the table provided to help manage the flow of work amongst human workers concerning assigning responsibilities. In the progression model, the workflow section indicates the tasks to be completed as grouped into scenes with the corresponding worker assigned to the task. Also, some circumstances require the system to be a worker and complete part of the workflow. Therefore the interaction with the application is partially captured as well.

For example, the worker named Jim may want to determine if another worker has completed their part of the progression. He can look at the workflow panel and see which worker is

responsible for a particular scene. He can also see what he is responsible for, such as in this scenario, where he is required to provide the account information for this transaction in the fourth scene. Figure 9 shows the worklist of workers assigned to scenes in the workflow panel.

Federated Workflow addresses the issue of how workflow systems interoperate. We have not investigated the implications of the progression model and Federated Workflow. Expressing workflow with a markup language may be conducive to integrating workflow systems since the workflow is stated explicitly.

Workflow

Scene	Name	Worker	Status
1	EnterName	Dave	Complete
2	EnterAddre...	Dave	In Progress
3	EnterCredit	Tara	Inactive
4	EnterAccou...	Jim	Inactive
5	Confirm	Jen	Inactive

Figure 9. The worklist in the workflow panel

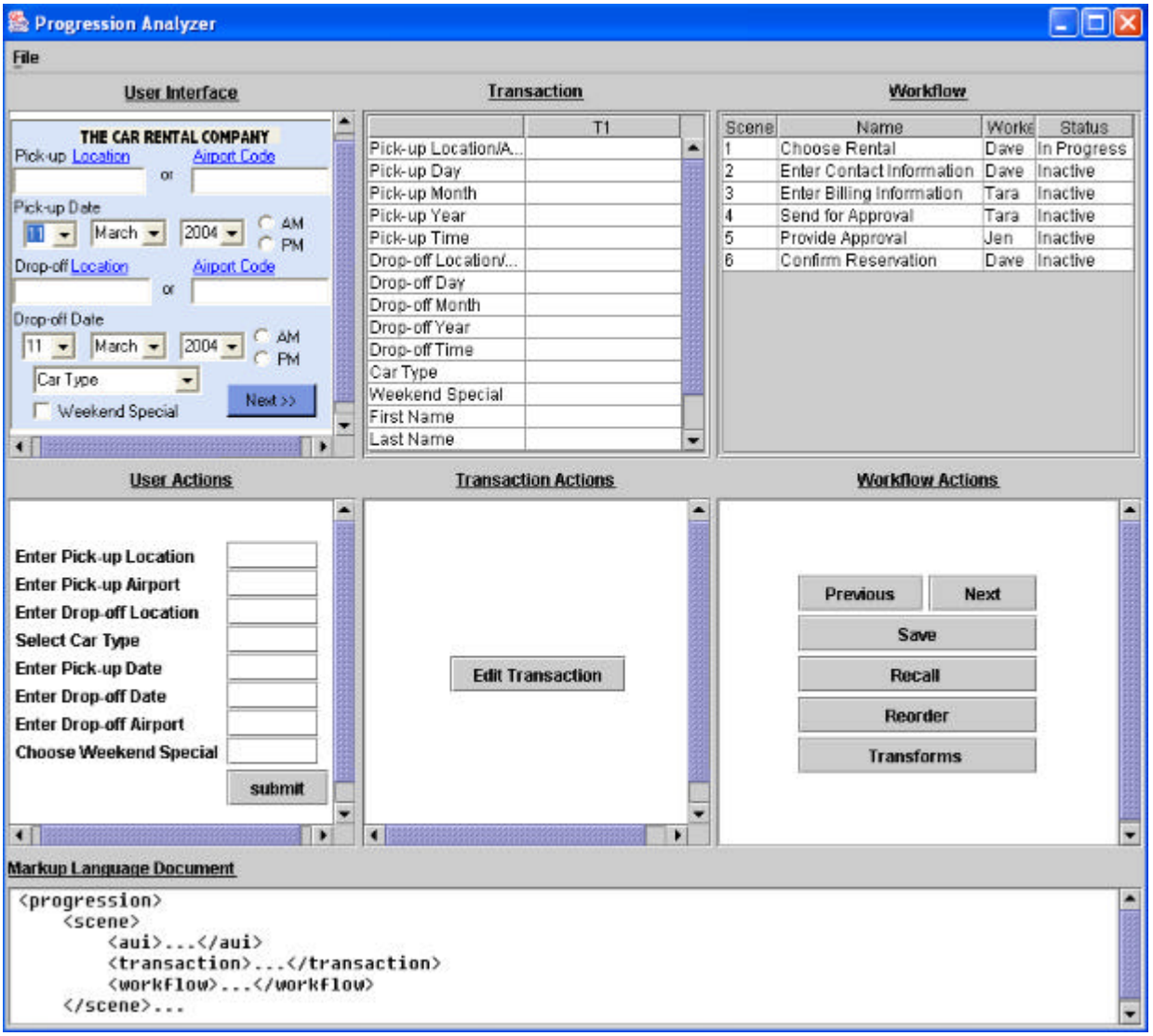

Figure 10. A screenshot of a more complex user interface and progression model in the Progression Analyzer.

CONCLUSION

User interface description languages have been designed to specify various aspects of a user interface, such as the tasks to be accomplished by the user, the abstract and concrete elements in the user interface and the user interface dialog. Workflow aspects, such as coordinating and managing tasks, are not modeled by current UIDLs. The progression model is an attempt to identify workflow issues

and to integrate workflow into a user interface description language.

We have developed a prototype system, the progression analyzer, to help refine the progression model and to investigate workflow in a UIDL. Explicitly recording and manipulating progressions allows us to dynamically change the workflow of an interactive system. Benefits include, improving the plasticity of an interactive system (workflow plasticity), providing a coarser integration with an application (transaction-based integration), and additional workflow functionality.

The progression analyzer has provided us a mechanism for studying progressions in detail. From this experience, we plan to formalize our XML-compliant language to show how workflow concepts can be integrated into a user interface markup language.

REFERENCES

1. Abrams, M., Phanouriou, C., Batongbacal, A.L., Williams, S., and Shuster, J. UIML: An Appliance-Independent XML User Interface Language. In A. Mendelson, editor, *Proceedings of 8th International World-Wide Web Conference WWW's* (*Toronto, May 11-14, 1999*), Amsterdam, 1999. Elsevier Science Publishers.
2. Annett, J., Duncan, K.D., Stammers, R.B., & Gray, M.J. (1971). Task analysis. London: Her Majesty's Stationery Office.
3. Arsanjani, A., Chamberlain, D., and et al. (WSXL) web services experience language version, 2002.
4. Azevedo, P., Merrick, R., and Roberts, D. OVID to AUIML – user-oriented interface modeling. In N. Nunes, editor, *Proceedings of 1st International Workshop "Towards a UML Profile for Interactive Systems Development" TUPIS'00* (York, October 23, 2000)., York, 2000.
5. Bernstein, A. How Can Cooperative Work Tools Support Dynamic Group Processes? Bridging the Specificity Frontier. (CSCW'00), 2000, pp. 279-288.
6. Card, S. K., Moran, T. P., and Newell, A., The Psychology of Human-Computer Interaction , Lawrence Erlbaum, 1983.
7. Haake, J., Wang, W. Flexible Support for Business Processes: Extending Cooperative Hypermedia with Process Support. (GROUP'97), 1997, pp. 341-350.
8. Hartson, H. R, Siochi, A. C., Hix, D. The UAN: a user-oriented representation for direct manipulation interface designs. ACM Transactions on Information
9. Heinl, P., Horn, S., Jablonski, S., Neeb, J., Stein, K., Teschke, M. A Comprehensive Approach to Flexibility in Workflow Management Systems. (WACC'99), 1999, pp. 79-88.
10. Luyten, K., Clerckx, T., Coninx, K., Vanderdonckt, J. Derivation of a Dialog Model from a Task Model by Activity Chain Extraction. (DSV-IS'2003), Funchal, Madeira Island (Portugal), 2003, ©Springer-Verlag 2003.
11. Luyten, K., Vandervelpen, C., and Coninx, K. Adaptable user interfaces in component based development for embedded systems. In *Proceedings of the 9th Int. Workshop on Design, Specification, and Verification of Interactive Systems DSV-IS'2002*, (Rostock, June 12-14, 2002). Springer Verlag, 2002.
12. Manolescu, D. An Extensible Workflow Architecture with Objects and Patterns. Chapter 4 in *Technology of Object-Oriented Languages, Systems, and Architectures* Theo D'Hondt, editor. Kluwer Academic Publishers, 2003.
13. Narendra, N. C., Adaptive Workflow Management – An Integrated Approach and System Architecture. (SAC'00), 2000, pp. 858-865.
14. Paternò, F., Mancini, C., Meniconi, S. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. (Proceedings Interact'97), Chapman&Hall, 1997, pp.362-369.
15. Paternò , F. and Santoro, C. One model, many interfaces. In Ch Kolski and J. Vanderdonckt (Eds.), editors, *Proceedings of the 4h International Conference on Computer-Aided Design of User Interfaces CADUI'2002* (Valenciennes, 15-17 May 2002), pages 143-154, Dordrecht, 2002. Kluwer Academic Publishers.
16. Puerta, A. and Eisenstein. XIML: A common representation for interaction data. In *Proc. Of the 7th International Conference on Intelligent User Interfaces* (Santa Fe, United States, January 2002), pages 69-76., New York, 2002. ACM Press.
17. Souchon, N. and Vanderdonckt, J. A Review of XML-Compliant User Interface Description Languages. (DSV-IS'2003), Funchal, Madeira Island (Portugal), 2003, ©Springer-Verlag 2003.
18. Stavness, N. and Schneider, K. A. Supporting Flexible Business Processes with a Progression Model, (IUI-CADUI 2004) Workshop: Making Model-based UI Design Practical: Usable and Open Methods and Tools , Island of Madeira, Portugal, January 2004
19. Traetteberg, H.: Modeling work: Workflow and Task modeling. In: Vanderdonckt, J., Puerta, A.R. (eds.): Proc. of 3 rd Int. Conf. on Computer-Aided Design of User Interfaces CADUI'99 (Louvain-la-Neuve, 21-23 October 1999). Kluwer Academics, Dordrecht (1999) 275–280.
20. WfMC. Workflow Management Coalition Terminology & Glossary, WFMC-TC-1011, Document Status- Issue 2.0, June 1996. Specifying Task Models. (Proceedings Interact'97), Chapman&Hall, 1997, pp.362-369