

Integrating task modelling into the object oriented design process: a pragmatic approach

Shijian Lu^{*}, *Cécile Paris*^{*} & *Keith Vander Linden*^{**}

^{*}*CSIRO/MIS, Locked Bag 17
North Ryde, NSW 2113
Australia
+61 2 9325-3152
{shijian.lu, cecile.paris }@cmis.csiro.au*

^{**}*Department of Computer Science
Calvin College
Grand Rapids, MI 49546
USA
kvlinden@calvin.edu*

Abstract

It is well recognised that task analysis and modelling are complementary to object oriented software development. While the latter has gained widespread use, the adoption of the former is still rare in industry. We argue that while good methodologies are important, they are not sufficient by themselves to ensure their rapid adoption. What is also needed are tools which will mediate the implementation process. In this paper, we propose a pragmatic approach to integrating task modelling into the object oriented design process. A task modelling tool— TAMOT is presented as part of our approach which uses TAMOT and OO CASE tools as pillars, and task models as a bridge to link task modelling and object oriented paradigms. The integration of TAMOT and CASE tools is achieved through two transformers, BTT and TTB. The BTT will automatically construct corresponding task models from system behaviour models while the TTB would do the reverse.

1. Introduction

The success of a system should be measured by how well it supports the users in accomplishing their tasks. It is not surprising, therefore, that task analysis and modelling were first employed in system evaluation. However, it has been realised that the usability of the system under development should be improved if task analysis and modelling are incorporated into its early design process.

The Object Oriented (OO) approach to software development has become the paradigm of choice. This is partly due to the fact that it has many virtues over traditional approaches: it is easier to maintain, and it supports higher degree of reuse because its structure is inherently decoupled. It is also due to the fact that OO CASE tools are available, which make it a much more viable approach. Yet, object oriented development is largely system centred, rather weak in capturing the users' usage and interaction patterns, all of which are critical to the useability of the final system from the user's perspective.

Task analysis and modelling, on the other hand, provide a user centred view. They are designed to describe users' tasks. Moreover, they allow for an assessment of the users' cognitive load and the validation of the resulting task model.

It is well recognised that task analysis and modelling provide OO with its much-needed strength. However, to integrate the two effectively in an industrial context, a proper mechanism has to be in place. One of the pre-requisites is to have a software tool for supporting task analysis and modelling. Another critical factor is how to feed the result of task modelling into subsequent system design (van Harmelen, 1997). In this paper, we put forward a pragmatic approach to integrate task modelling into the industrial OO software development process. This approach is built on top of an OO CASE tool—Rational ROSE and a task modelling tool—TAMOT. Supporting Diane+ (Tarby, & Barthet, 1996), a formal graphical notation for describing task models, TAMOT is integrated with ROSE through two transformers, BTT (system Behaviour model To Task model) and TTB (Task model To system Behaviour model) which exploit the semantic common ground between system behaviour models and task models. The BTT will automatically construct corresponding task models from system behaviour models while the TTB would do the reverse. This provides a mechanism to feed the result of task modelling into ROSE for subsequent design.

2. Our approach

Our approach can be illustrated by Figure 1. During requirement analysis, designers, HCI specialists and end users work together to conduct task analysis (Lim, 1996, Wilson & Johnson, 1996) which roughly involves the following stages:

- Create coarse grained task model
- Construct medium-grained task model
Tasks in the task model derived earlier are further decomposed into sub-tasks till the level of detail where the task is identified as either performed by the user or the system.
- Construct fine-grained task model (interaction task model)
The task model is further decomposed into elementary task level, the lowest level possible. By then the detailed interaction between the user and the system is un-ambiguously specified.

The result of task analysis is a task model that is captured by a task-modelling tool, TAMOT. At any stage of task analysis, the resulting task model could be fed into ROSE through the task model to system behaviour model transformer, TTB which would automatically generate a draft system behaviour model (refer to next section). Of course, the level of details of the generated system behaviour model will reflect its corresponding task model.

In the CASE tool, a structural model of the systems behaviour will be created by software engineers. At the same time, the system behaviour model will be refined. Interaction diagrams, in particular, will be fleshed out as they contain few system actions when first generated (refer to next section).

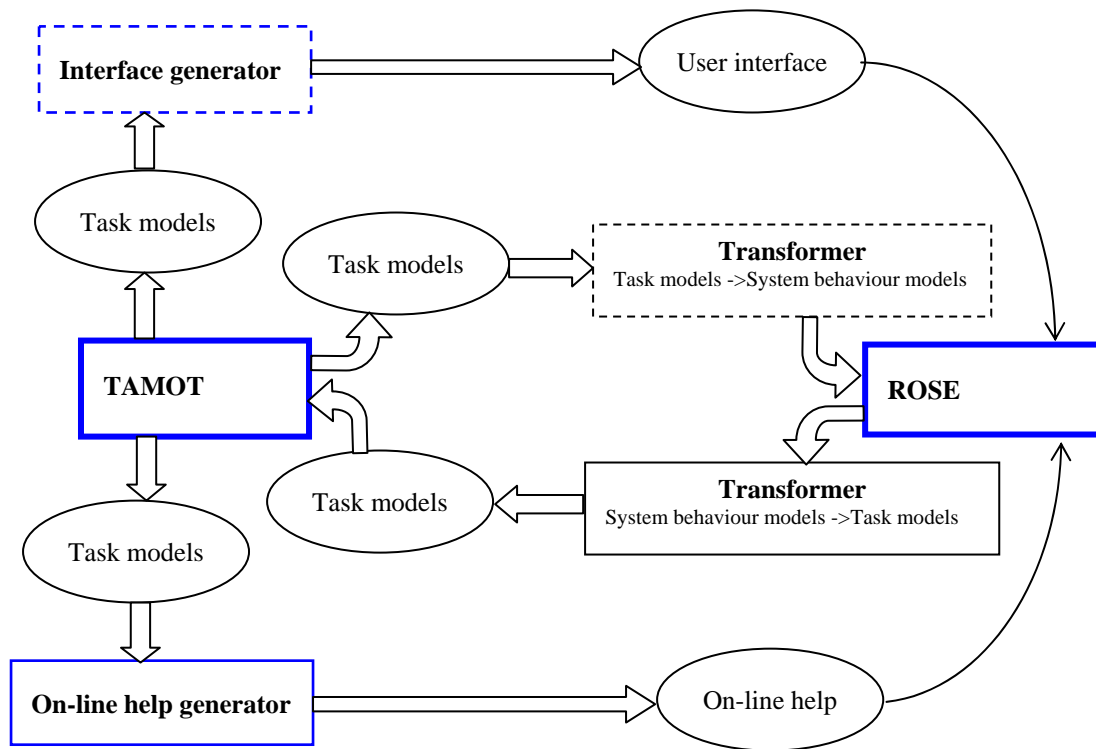


Figure 1 The schematic diagram of the proposed approach

It is up to the development team to decide at what stage of task analysis to feed the task model to the CASE tool. They may choose to do that in the early stage of task analysis, and subsequently, work on the resulting system behaviour model. If this is the case, the improved system behaviour model would be used to construct corresponding task model through the BTT. Residing in ROSE, BTT was built in ROSE script. Therefore, it does not matter whether improvement is made on the task model or the system behaviour model, the consistency between the two is largely ensured automatically. At any stage of development, on one hand, source codes for the system under development are generated in ROSE, and on another hand, the task model in TAMOT could be tuned to generate user interface codes and/or user documentation. These (i.e., system codes, interface codes and on-line help) are then combined to form a working prototype system with documentation. This approach thus supports incremental and iterative development.

3. System behaviour models versus task models

System behaviour models in an OO CASE tool (e.g., Rational Rose) typically include use cases (Jacobson, et al, 1995, Pressman, 1997), use case diagrams, interaction diagrams, and state transition diagrams (UML, 1997). Among them, state transition diagrams are rarely used because they tend to be too difficult to build. We will, therefore, focus on use cases, use case diagrams, and interaction diagrams.

It has been recognised that there is a strong similarity between use cases and task models (Artim 1997). In fact, not only use cases, but also use case diagrams and interaction diagrams are semantically similar to task models in the following major ways (Lu, Paris, & Vander Linden, 1998).

- Use cases and use case diagrams can be seen as equivalent to composite tasks in a task model;
- The containment of use cases by other user-case diagrams resembles the hierarchical structure in a task model;
- Extend relationships in use case diagrams can be seen as equivalent to task sequence preconditions;
- In terms of scope, task models are a sub-set of interaction diagrams, but they contain considerably more information on user tasks;
- While some task attributes are defined explicitly, others are defined implicitly, and some are not defined at all in system behaviour models.

Given the fact that system behaviour models and task models resemble each other to such a large degree and the fact that large part of task models can be automatically constructed from system behaviour models (Lu, Paris, & Vander Linden, 1998), we are looking at the possibility of automatically building use cases, use case diagrams and interaction diagrams from task models. It is envisioned that use cases and use case diagrams would be constructed from the high-level tasks in a task model. One would also expect that only a skeleton interaction diagram could be automatically acquired from task models due to the fact that interaction diagrams are wider in scope. That is, while task models mainly describe users' and system's overt behaviour, interaction diagrams, in addition to that, also cover the system's internal behaviour. Consequently, the resulting interaction diagrams would contain few system events. Even so, that would provide a good starting point to software engineers to follow on with the rest of system design.

4. The graphical task modelling Tool: TAMOT

There are many formal notations for representing task models (Scapin & Pierret-Golbreich, 1990, Hix & Hartson, 1993, Tarby, & Barthet, 1996). Software tools for facilitating task model building, however, are scarce. It is fine to build task models manually for research purposes. But that is not good enough if we are serious about industrial deployment of task modelling. Having troubles finding existing task model building tools, we built TAMOT ourselves. Using the Diane+ notation, TAMOT is very much like a diagram editing tool within a CASE tool. The user interface of TAMOT, shown in Figure 2, was designed based on a user requirement analysis (Paris, Balbo, & Ozkan, 1997). With TAMOT, a task can be created and its attributes, such as automatic, manual or interactive; mandatory or optional; elementary or composite; precondition and feedback etc, modified easily. Task models can be created by incrementally by decomposing sub-tasks. However, TAMOT is more than an ordinary diagram tool in that it is capable of importing draft task models extracted from a CASE tool such as ROSE. Furthermore, it can export task models in such a way that they can be used by a language generator to create on-line help (Paris, Vander Linden, & Lu).

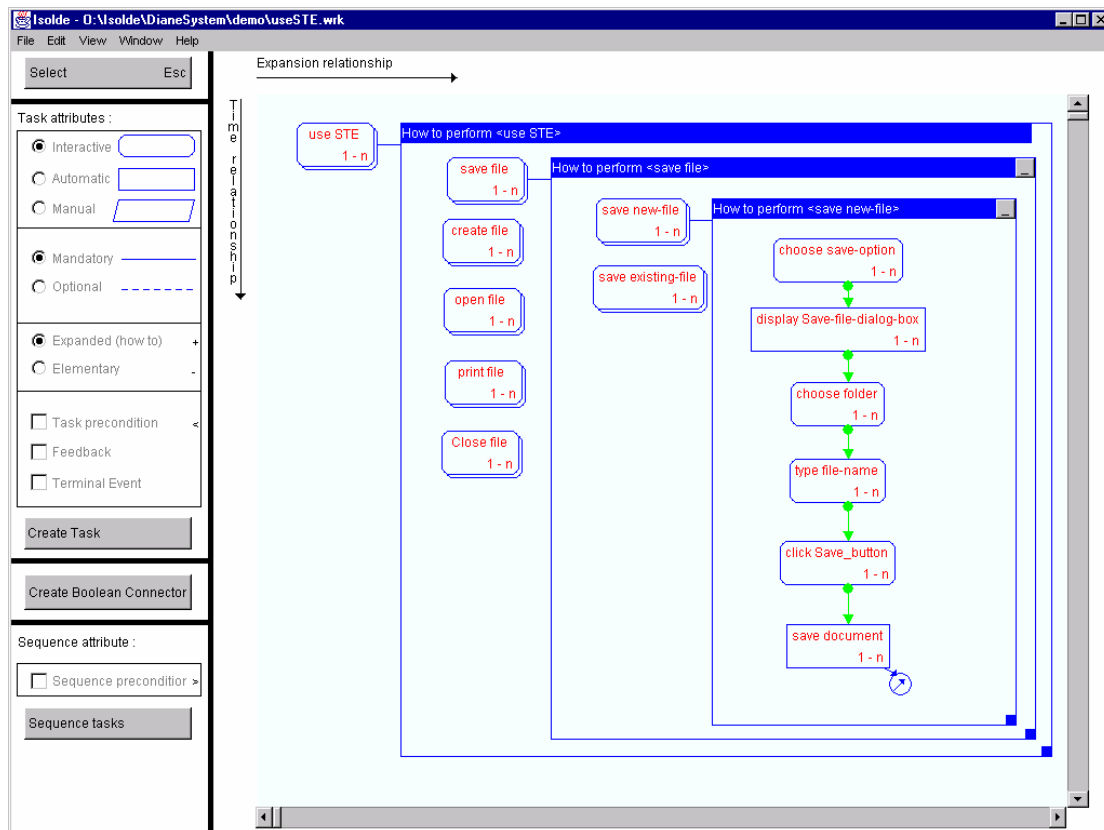


Figure 2 The user interface of the task-modelling tool: TAMOT

5. Discussion

In this paper, we briefly presented a pragmatic approach to incorporate task modelling into OO design process. During requirement analysis, instead of conventional OO use–case analysis task analysis is conducted. Naturally, the result of analysis will be task models instead of system behaviour models. The resulting task model will be formerly represented and captured in a task-modelling tool. It could then be used to automatically generate use cases and use case diagrams, even skeleton interaction diagrams in a case tool. Therefore, task modelling tools, such as TAMOT, not only provide much needed support to incorporating task modelling into OO analysis phase, but also into system design phase. Furthermore, it also provides a platform to incorporate user interface and on-line help generation into OO development.

The reason it is called a “pragmatic” approach is because it based on existing tools, i.e., TAMOT and especially an industrial strength CASE tool--Rational ROSE. Consequently, this approach is easier to adopt in practice. A more radical approach would be to truly integrate not only task modelling tools but also interface generator and on-line help generator with existing OO CASE tools into one super-tool. In this case, use case and use case diagrams would be completely replaced by task models and interaction diagrams would then be directly attached to task models. With such a tool, it would be possible to keep task models, interaction diagrams and user interface synchronised. That means that modification of task models would be automatically reflected in interaction diagrams and user interface, and vice versa. We leave this for future work.

Acknowledgments

This work is partially supported by the Office of Naval Research (ONR) – Grant N00014096-1-0465 – in the program for User Centred Direct Interaction Systems. We gratefully acknowledge the participation of all the members of our project team: Sandrine Balbo and Nadine Ozkan, and are thankful to Valery Anciaux and Christophe Plier for their contribution to the implementation of the task model editor.

References

- Artim, J.M. (1997) Integrating user interface design and object-oriented development through task analysis and use cases. CHI'97 workshop on Object Oriented User Interfaces. <http://www.cutsys.com/ooui/>
- Hix, D & Hartson, R (1993) *Developing user interfaces, ensuring usability through product & process*. Wiley.
- Jacobson, I, Christerson, M, Jonsson, P. and Overgard, G. (1995) *Object oriented software engineering: a use case driven approach*. Menlo Park, California, Addison-Wesley.
- Lim, K.Y. (1996) Structured task analysis: an instantiation of the MUSE method for usability engineering. *Interacting with computer*, Vol.8, No.1 pp31-50.
- Lu, S., Paris, C., Vander Linden, K. (1998) Towards the Automatic Construction of Task Models from Object-Oriented Diagrams. Submitted.
- Paris, C., Balbo, S. & Ozkan, N. (1997) Novel uses of task models: two case studies. *Proc. NATO/ONR workshop on cognitive tasks*.
- Paris, C., Vander Linden, K., Lu, S. (1998) A practical approach to the generation of on-line help. Submitted.
- Pressman, R.S. (1997) *Software engineering: a practitioner's approach*. International Edition, McGraw-Hill.
- Scapin, D. & Pierret-Golbreich, C. (1990) Toward s method for task description: MAD. Proc. Work with display units 89. Berlinquet, L. & Berthelette, D. (eds), Elsevier science publisher B.V.
- Tarby, J-C and Barthet, M-F (1996) the Diane+ method. *Proc. Second international workshop on computer-aided design of user interfaces*. Namur, Belgium, June.
- UML (1997) Notation Guide: unified modelling language version 1.0, Rational Software corporation.
- Van Harmelen, M. (1997) Idiom: an object-oriented user interface design methodology. CHI'97 workshop on Object Oriented User Interfaces. <http://www.cutsys.com/ooui/>
- Wilson, S. & Johnson, P. (1996) Birdge the generation gap: from work tasks to user interface design. Proc. CADUI'96, Vanderdonckt, J. Eds.

Brief biography

Shijian Lu is a research scientist in the Intelligent Interactive Technology Group at the Division of Mathematical and information Sciences of CSIRO (Commonwealth Scientific and Industrial Research Organisation). Prior to that, he worked for the Centre for Cognitive Science, RUC, Demark, as a research associate. He has a PhD degree from Leeds University, UK.