Providing automated support for building Dynamic Model based on Use Case Model

Hidekazu Yamasaki

School of Information Science,
Japan Advanced Institute of Science and Technology

February 15, 2001

Keywords: object oriented development, use-case, MSC, ObTS, ObCL.

Generally, software system development is a complex task. Several different aspects must be taken into consideration for development of reliable system that runs properly. But system development is so complex that we cannot cope well with handling many requirements at the same time. In order to manage with this complexity, methods using several different models in a system development have been introduced so far.

All systems have a *static structure* and *dynamic behavior*. *Dynamic model* gives expression to the behavior and is widely used to specify how objects perform interaction. A dynamic model gives expression state transition diagrams which is consist of *events*, *states* and *state transitions* in a object. A dynamic model corresponds to a state diagram in the Unified Modeling Language(UML).

Required functionalities for a system is defined by *use case model*. Use case model consists of a *system border*, *actors* and *use cases*. A use case defines a functionality and consists of one or some scenarios described by natural language. Use case model controls the formation of all other models at each process, i.e. analysis, design, implementation and test. The reason is the all models must satisfy requirements of functionalities defined by use cases. This method is called *use case driven design*.

Actually, we repeatedly make modifications of requirement specification for a system by way of adding a new functionality and/or modifying a functionality in use case model. Because all the models should be builded based on the use case driven design throughout the system development, we must modify each models with modification of the use case model.

Therefore, dynamic model is naturally modified based on use case model. But, it is not efficient that whenever a use case model is modified, we must modify a dynamic model based on the use case model.

In this paper, we aim to implement support environment that generates from a use case model to a dynamic model and introduce a method to build a dynamic model with this environment. By applying the method, we can immediately reflect modifications of a use case model to a dynamic model and build a dynamic model flexibility and efficiently.

Dynamic model represents interactions between objects with events and states. So, our approach is that we extract objects and message sequences between them from use case model, and then build dynamic model automatically.

Scenarios describe event flow to define a functionality in a use case. So we can extract message sequences without difficulty. But it is difficult to extract message sequences from scenarios with computation because they are described by natural language. Therefore, we suggested a formal description with sequence chart which is general model to describe message sequences in a use case.

ITU standard language Message Sequence Chart(*MSC*) is a widespread means for a message sequence specification in the communication field. The MSC has a formal graphical and a formal textual syntax. The MSC has structural concepts concerning structuring and modularization. Therefore, we decided to describe formally message sequences with the MSC and redefined the MSC to give a detail description which represents message sequences structure and object behavior in message communication.

In our research, we use *ObTS* and *ObCL* to describe dynamic model. ObTS is dynamic model which has the character of a readable and a reuse-able. ObCL defines concrete syntax for ObTS model. We implement MSC to ObCL converter which converts from MSC code to ObCL code. *ObML* is the simulator constructed on Standard ML. The ObCL code which describes a system is converted to ML code by ObCL converter, and these codes can be simulated and tested on ObML simulator. The results of the tests are feed back to the use case and MSC, and can be used to refinement at analysis and design process.

Finally, we proposed the method that we can consistently develop dynamic model from use case model with the support environment that we implemented. By applying the method to several examples, we confirmed that the method is effective and flexible.