Chapter VII

Rational Unified Process and Unified Modeling Language
A GOMS Analysis

Keng Siau
University of Nebraska-Lincoln, USA

GOMS is a model that analyzes knowledge of how to do a task in terms of Goals, Operators, Methods and Selection rules. GOMS is one of the most popular theoretical models in the field of human-computer interaction. Since its introduction, the GOMS model has been extended, enhanced and applied to areas outside human-computer interaction. The goal of this chapter is to discuss the use of the GOMS model for the design and evaluation of modeling techniques. In this chapter, we introduce the GOMS concepts, discuss the applicability of GOMS for modeling and describe how GOMS can be used to analyze Rational Unified Process and Unified Modeling Language.

INTRODUCTION

The Unified Modeling Language (UML) is a visual modeling language for modeling system requirements, describing designs and depicting implementation details. Grady Booch, Jim Rumbaugh and Ivars Jacobson, known collectively as the “three Amigos” at Rational Software Corp, spearheaded development of UML in the mid-1990s. Unified Modeling Language (UML) borrows concepts from a large number of different methodologies, and is tailored specifically for object-oriented design. Since its inception, UML has emerged as the software industry’s dominant modeling language. UML is not only the de facto modeling language standard for specifying, visualizing, constructing and documenting the components of software systems but it is also fast becoming a de jure standard (Booch, 1999).

By offering a common blueprint language, UML relieves developers of the proprietary ties that are so common in this industry. Major vendors, including IBM, Microsoft and Oracle, are brought together under the UML umbrella. Many of the language’s supporters claim that UML’s simplicity is its chief benefit (Kobryn, 1999) and argue that UML uses simple, intuitive notations that are understandable by nonprogrammers. If developers,
customers and implementers can all understand a single modeling language instead of a few
dozens (Siau, 1999; Siau and Cao, 2001), they are more likely to agree on the intended
functionality, thereby improving the communication process among the stakeholders and
enhancing their chances of creating an application that truly addresses business problems.

Rational Unified Process

The Rational Unified Process is a software engineering process developed and
marketed by Rational Software. It uses UML when preparing all blueprints of the software
system (Jacobson et al., 1999). UML is considered to be an integral part of the Rational
Unified Process and the two were developed hand in hand. The Rational Unified Process,
in a nutshell, consists of four main concepts it is use-case driven, architecture-centric,
iterative and incremental. Major goals of the Rational Unified Process are model develop-
ment and maintenance for systems under development (Krutchen, 2000).

Unified Modeling Language

The Unified Modeling Language (UML) defines a number of graphical views that
provide different perspectives of the system under development. Each diagram shows a
different aspect of the full model, and is by design incomplete. UML encompasses a total
of nine views, which taken together, form a comprehensive model of the system. Some of
the views depict the static aspect of the system whereas others show the dynamic aspect. The
views include class diagrams, use-case diagrams, statechart diagrams, activity diagrams,
sequence diagrams, collaboration diagrams, object diagrams, components diagrams and
deployment diagrams (Booch et al., 1999). A class diagram shows a set of classes,
interfaces, and collaborations and their relationships. An object diagram depicts static
“snapshots” of the elements within a system, showing objects’ structure, attributes and
relationships to one another. An activity diagram shows the flow of control from one
activity to the next, and a use-case diagram illustrates how elements outside the system use
the system. For instance, the internal workings of a new payroll system would be shown in
an activity diagram, whereas external actors, such as the mail order department, would
appear in a use-case diagram. Sequence and collaboration diagrams show interactive
processes: developers see not only objects and classes, but also the messages that pass
between them. Thus, developers can simulate passes through the system using a conven-
tional “what if” approach. A statechart diagram shows a state machine, consisting of states,
transitions, events and activities. Finally, component and deployment diagrams show the
physical or implementation view of the system (including executables, libraries and
interfaces).

In this research, we are interested in analyzing the Rational Unified Process and UML
using GOMS. The rest of the chapter is organized as follows: the next section discusses the
GOMS model and the components in the model. The applicability of GOMS to studying
modeling techniques is then discussed. Finally, we apply GOMS to analyze the Rational
Unified Process and UML.

THE GOMS MODEL

The GOMS model was introduced in the seminal book, The Psychology of Human-
Computer Interaction by Card, Moran and Newell (1983). They suggested that, when
designing user interfaces, it is helpful to analyze operations on how to perform a task in terms
of four components: a set of goals, a set of operators, a set of methods1 for achieving the
Related Content

Developing Rule-Based Web Applications: Methodologies and Tools
www.irma-international.org/chapter/developing-rule-based-web-applications/35867/

New Dimension in Relational Database Preservation: Using Ontologies
www.irma-international.org/chapter/new-dimension-relational-database-preservation/73178/

XML and LDAP Integration: Issues and Trends
www.irma-international.org/chapter/xml-ldap-integration/27776/

A Secure and Dynamic Mobile Identity Wallet Authorization Architecture Based on a XMPP Messaging Infrastructure
www.irma-international.org/chapter/secure-dynamic-mobile-identity-wallet/73171/

Complexity-Based Evaluation of the Evolution of XML and UML Systems
Ana Isabel Cardoso, Peter Kokol, Mitja Lenic and Rui Gustavo Crespo (2005). Advances in UML and XML-Based Software Evolution (pp. 308-321).
www.irma-international.org/chapter/complexity-based-evaluation-evolution-xml/4941/