Developing Complex Systems with Object-Process Methodology Using OPCAT

Dov Dori, Iris Reinhartz-Berger, and Arnon Sturm

Technion, Israel Institute of Technology
Technion City, Haifa 32000, Israel
Emails: {dori@ie, ieiris@tx, sturm@tx}.technion.ac.il

Abstract. OPCAT—Object-Process CASE Tool—is an integrated systems engineering environment. It supports system lifecycle evolution using Object-Process Methodology (OPM). OPM integrates the object-oriented (structure) and process-oriented (behavior) paradigms into a single frame of reference through a combination of graphics and equivalent natural language. This short paper briefly describes OPM and demonstrates highlights of OPCAT and some of its capabilities.

1 The Basis: Object-Process Methodology

Object-Process Methodology (OPM) [1] is a holistic approach to the study and development of systems. It integrates the object-oriented and process-oriented paradigms into a single frame of reference. Structure and behavior, the two major aspects that each system exhibits, co-exist in the same OPM view without highlighting one at the expense of suppressing the other. The elements of the OPM ontology are entities (stateful objects and processes) and links. Objects are (physical or informatical) things that exist, while processes are things that transform objects. Links can be structural or procedural. Structural links express static relations between pairs of entities. Procedural links connect entities to describe the behavior of a system. The behavior is manifested in three major ways: processes can transform objects, objects can enable processes, and objects can trigger events that invoke processes.

Two semantically equivalent modalities, one graphic and the other textual, jointly express the same OPM model. A set of inter-related Object-Process Diagrams (OPDs) constitute the graphical, visual OPM formalism. Each OPM element is denoted in an OPD by a symbol, and the OPD syntax specifies correct and consistent ways by which entities can be linked. The Object-Process Language (OPL), defined by a grammar, is the textual counterpart modality of the graphical OPD-set. OPL is a dual-purpose language, oriented towards humans as well as machines. Catering to human needs, OPL is designed as a constrained subset of English, which serves domain experts and system architects engaged in analyzing and designing a system. Every OPD construct is expressed by a semantically equivalent OPL sentence or phrase. Designed also for machine interpretation, OPL provides a solid basis for automatically generating the designed application. This dual representation of OPM increases the processing capability of humans. Another advantage of OPM is its complexity management mechanisms. OPM offers three refinement/abstraction

mechanisms: (1) unfolding/folding is used for refining/abstracting the structural hierarchy of a thing; (2) in-zooming/out-zooming exposes/hides the inner details of a thing within its frame; and (3) state expressing/suppressing exposes/hides the states of an object. Using flexible combinations of these mechanisms, OPM enables specifying a system to any desired level of detail without losing legibility and comprehension of the resulting specification. The complete OPM system specification is a set of OPDs and their corresponding OPL paragraphs.

2 OPCAT Overview

Based on human cognition principles, OPCAT [2, 3] implements OPM and enables bimodal visual-lingual balanced modeling of the structural and behavioral aspects of systems in a single view. Due to this intuitive dual notation, the resulting model is comprehensible to both domain experts and system architects engaged in the development process. Due to OPM formality, OPCAT also provides a solid basis for implementation generation and an advanced simulation tool, which animates system behavior. OPCAT enables generic translation of the OPL (subset of English) script to various formal target languages. Currently we generate Java code from OPL.

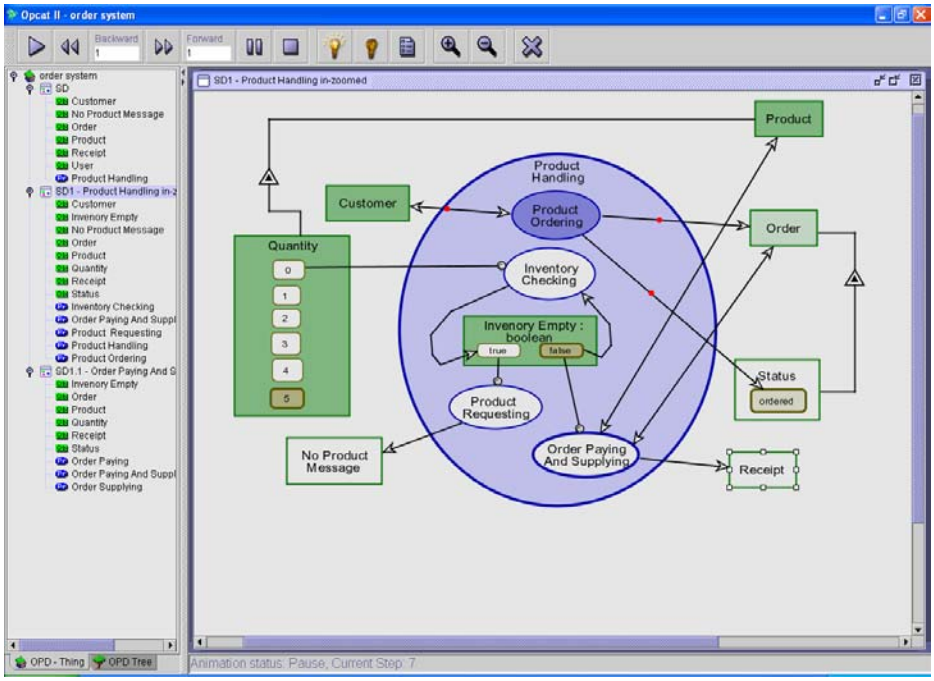

Figure 1. A snapshot of OPCAT simulating an inventory system

Figure 1 is a snapshot of OPCAT simulating an inventory system. The OPD shows in dark green objects that already exist (e.g., **Customer**, **Product**), while light green, becoming dark, (e.g., **Order**) represents objects being generated. Dark blue is the

process currently in action (**Product Ordering**), while light blue (**Product Handling**) is the higher-level process of which the current process is subprocess. The red dots indicate the progress of the control. For example, the red dot along the arrow (result link) from **Product Ordering** to **Order** indicates that **Product Ordering** is about half done so **Order** is about half ready.

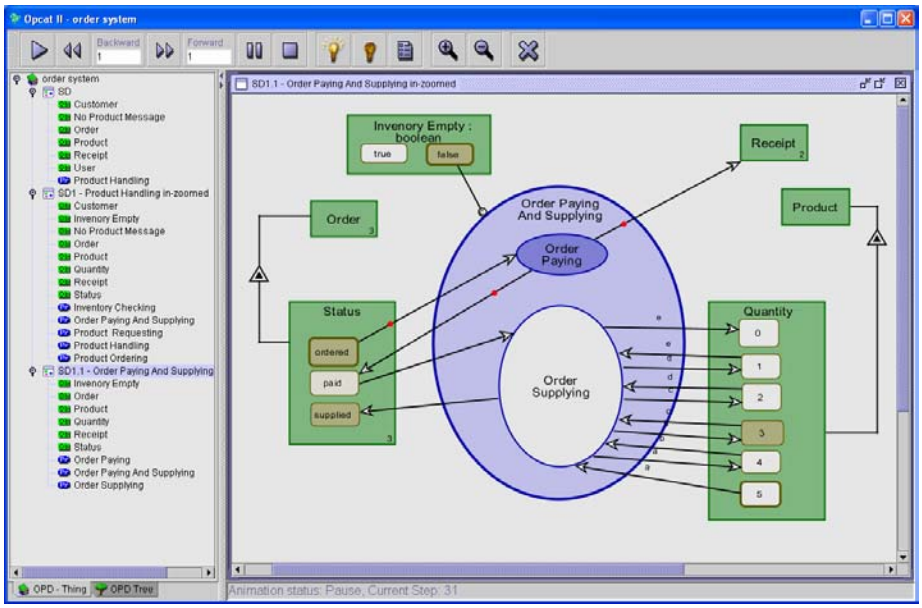

Figure 2. The process Order Paying and Supplying in-zoomed

Figure 2 shows the process **Order Paying and Supplying** of Figure 1 in-zoomed, exposing its two subprocesses, **Order Paying** and **Order Supplying**. Currently, **Order Paying** is being executed and it changes the **Status** attribute of **Order** from **ordered** to **paid**.

Other prominent OPCAT features beside code generation include the generation of UML diagrams from the OPM specification as well as automated documentation generation in various parameter-governed formats. Projects under way include collaborative capability, automated diagram layout, Visual Semantic Web, and reverse engineering.

References

1 Dori, D. Object-Process Methodology - A Holistic Systems Paradigm, Springer Verlag, Berlin, Heidelberg, New York, 2002.

2 Dori, D. Reinhartz-Berger, I. and Sturm A. *OPCAT* – A Bimodal Case Tool for Object-Process Based System Development. 5th International Conference on Enterprise Information Systems (ICEIS 2003), pp. 286-291, 2003.

3 OPCAT download site: http://www.ObjectProcess.org