Object Process Methodology as a Business Process Modeling Tool

Dov Dori
Massachusetts Institute of Technology

Follow this and additional works at: http://aisel.aisnet.org/ecis2000

Recommended Citation
http://aisel.aisnet.org/ecis2000/172

This material is brought to you by the European Conference on Information Systems (ECIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ECIS 2000 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
Object-Process Methodology as a Business-Process Modeling Tool

Dov Dori
Sloan School of Management
Massachusetts Institute of Technology
and
Faculty of Industrial Engineering and Management
Technion, Israel Institute of Technology
Haifa 32000, Israel

Abstract Object-Process Methodology (OPM) is a system development and specification approach that combines the two aspects of process, structure and behavior, within a single graphical and textual model. This paper specifies an electronic commerce system in a hierarchical manner. At the top are processes of managing a generic product supply chain before and after the product is manufactured. Focusing on the post-product supply chain management, we gradually refine the details of the fundamental, almost classical electronic commerce interaction between the retailer and the end customer, namely payment over the Internet using the customer's credit card. The specification results in a set of Object-Process Diagrams and a corresponding equivalent set of Object-Process Language sentences. The synergy of combining structure and behavior within a single formal model, specified both graphically and textually, yields a highly expressive system modeling and specification tool. The comprehensive, unambiguous treatment of this basic electronic commerce process is formal, yet intuitive and clear, suggesting that OPM is a prime candidate for becoming a common standard vehicle for defining, specifying and analyzing systems.

1 INTRODUCTION

Time-to-Market and Time-to-Revenue are factors of growing importance in new product development and marketing. At the same time, products and systems are becoming ever more complex, making their specification, analysis, design and implementation more difficult. These complex systems require systematic development and support efforts throughout their entire lifetime.

Current definitions of electronic commerce include the ability to conduct business via electronic networks, or the term used to describe the wide range of tools and techniques utilized to conduct business in a paperless environment [1]. According to Reeling [2], electronic commerce has become a major focus for many organizations and represents a significant opportunity for integrated supply management efforts. As Pine [3] notes, product development, manufacturing and supply chain management have traditionally been thought of as separate business processes. He advocates that they be considered in tandem as a three dimensional consumer engineering approach. This paper proposes that electronic commerce is a business process that enables these three lifecycle activities.

Since electronic commerce and supply chain management are complex enough each on its own, a generic, sound methodology that is capable of modeling both the physical and informational facets of the combined system is essential. This paper proposes the Object-Process Methodology (OPM) as a prime vehicle for modeling electronic commerce and related complex systems. Such systems feature physical and informational aspects while exhibiting rich structural and dynamic features. The paper first introduces the principles of OPM and its applications. Using OPM we then model an electronic commerce mediated retailing-purchasing activity through the use of a credit card. The expressive power of the resulting workflow-like model stems from integrating structure and behavior within one model and representing them both graphically and textually. The graphic representation is provided as a set of Object-Process Diagrams (OPDs). The textual equivalent specification is in the form of a collection of sentences in Object-Process Language (OPL), a natural language derived automatically from the diagrams according to the syntax of a context free grammar that produces legal English sentences. The model is therefore instrumental in analyzing cause and effect and identifying possibilities for further system development, such as converting currently manual operations into automated electronic commerce operations.

2. THE OBJECT-PROCESS METHODOLOGY

The conventional wisdom among developers and users of systems development methodologies has been that there is an inherent dichotomy between object-oriented and process-oriented approaches, and that it is not possible to combine these two essential aspects of any system into one coherent model that provides an integral frame of reference. This misconception has accompanied systems development throughout, to the extent that all the important object-oriented systems analysis and design methodologies advocate the use of a number of models to specify and design a single system. As a result of this trend, current methodologies for information systems development suffer from a number of crucial shortcomings:

Lack of explicit process modeling: Limited by the capabilities of programming languages and conceived in a bottom-up fashion, they are object-oriented (OO), rather than object-process-oriented. In reality, more often than not, processes are at least as important and central to the system as are objects, and their occurrence involves a number of objects. In particular, electronic commerce is a classical instance of something that is a process rather than an object.

The model multiplicity problem: Current methodologies for information systems development cope with complexity management through aspect-based decomposition, leading to the use of a number of models to describe the same system. This yields the model multiplicity problem, which refers to the need to comprehend and mentally integrate a variety of models of the same system and constantly take care of synchronizing among them. This problem arises for any approach that tries to overcome the inherent complexity of a system by decomposing it into a number of models, each handling a different aspect (function, structure, behavior) using a different diagramming tool.

The currently accepted standard of the Unified Method Language, UML [4] unfortunately maintains the separation between
structure and behavior in separate models. UML advocates using eight different models. As we have shown by experimenting with OPM and OMT for specifying real-time systems [5], model multiplicity is a real problem. OPM solves it by using a single modeling tool.

OPM [6], [7] is a systems development approach that caters to the natural train of thought humans normally apply while trying to understand and build complex systems. In such systems, it is usually the case that structure and behavior are intertwined so tightly that any attempt to separate them is bound to further complicate the already complex description.

OPM incorporates the static-structural and behavioral-procedural aspects of a system into a single, unifying model. Founded on General Systems Theory, OPM generically models the structure and behavior of physical and informational things. It is therefore suitable for specifying systems of virtually any domain, e.g., R&D activities [8]. In particular, as we show in this paper, modeling electronic commerce systems with OPM provides a complete and concise system specifications.

OPM achieves model integration by incorporating the two major system aspects, structure and behavior, into a single model, in which both objects and processes are adequately represented without suppressing each other. As argued, this approach counters contemporary object-oriented systems development methods, which require several models to completely specify a system.

In the OPM ontology, objects are viewed as persistent state preserving things (entities) that interact with each other through processes. Processes are transient, transforming entities that affect objects by changing their state, or by generating or consuming objects. Separating structure from behavior while engaging in system modeling is counter-intuitive. The synergy of structure-behavior unification within a single model results in a highly expressive modeling tool.

3. OBJECT-PROCESS DIAGRAMS (OPD's)

OPM uses Object Process Diagrams (OPD's), drawn using OPCAT [9] for expressing the objects of a modeled system and the processes that affect them. OPCAT responds to some of the challenges that current CASE tools face [10]. The OPD's are elaborate workflow-like hypergraphs that model the system or parts of it at various levels of detail. We present OPD notations and symbols gradually as we model a supply chain management and the electronic commerce processes associated with it. Figure 1 shows the OPCAT graphic user interface with a top level OPD of a generic supply chain and the pertinent electronic commerce systems. The supply chain model is based on [1]. As the text next to the various symbols in Figure 1 shows, objects and processes are symbolized in an OPD as rectangles and ellipses, respectively. Pre-Product Supply & Production and Post-Product Supply & Retailing are the two processes in the OPD of Figure 1. The first process pertains to the entire suite of engineering, technological and commercial activities that take place before the End Product object is at hand. It is within this large process that Business-to-Business electronic commerce processes take part. The object Business-to-Business Electronic Commerce System in the OPD of Figure 1 is the instrument that enables it. The second process analogously concerns the set of transporting, distributing and retailing processes that End Product undergoes all the way to the end customer. Within this process, business-to-business and retailer-customer electronic commerce processes are executed.

Figure 1. A top-level Object-Process Diagram (OPD) of a generic supply chain system

Procedural links

Objects and processes are connected by procedural links, which can be either enabling links or transformation links. These two different kinds of links are used in the OPD of Figure 1 to connect objects to processes, depending on the roles that these objects play in the process to which they are linked. Objects may serve as enablers, which can be instruments or intelligent agents. They are involved in the process without changing their state. Objects may also be transformed (change their state, generated, consumed, or affected) as a result of a process acting on them. An enabling link connects an enabler to the process that it enables. Enabler is an enabling object that needs to be present in order for the process to occur but it does not change as a result of the process occurrence. An enabling link can be an agent link or an instrument link. An agent link is denoted by a line with a black circle at the process end, such as the one from the object Retailer to the process Post Product Supply & Retailing. An agent link denotes that relative to the enabled process, the agent is an intelligent agent, i.e., a human or an organizational unit that comprises humans, such as a department or an entire enterprise. An instrument link is an enabling link that is symbolized as a white circle at the process end, which denotes that the enabler is an instrument.

(a) (b) (c)

Figure 2. Object states and effect links. (a) The process Retailing affects the object Product by transforming it from state manufactured to state retailed. (b) States of Product suppressed. (c)
4. THE OBJECT-PROCESS LANGUAGE

Based on a constrained context-free grammar, a textual specification in a natural-like Object-Process Language (OPL) can be automatically extracted from the diagrammatic specification in the OPD set. Devoid of the idiosyncrasies and mess of cryptic details that feature current programming languages, OPL sentences are understandable to humans with prior training, and can therefore be a prime candidate for coming a business language for electronic commerce. OPL has two goals. One is to convert the set of Object-Process diagrams (OPDs) into a natural-language-like text for use as means for communicating analysis results back to domain experts, prospective users and customers, who may be more comfortable with reading text than checking Object-Process diagrams. The other goal of OPL is to provide the infrastructure needed for continuing the application development code generation and database scheme generation. These traits make the combination of the graphically-oriented OPD and its equivalent text-based OPL counterpart ideal infrastructure for systems specification in the domain of electronic commerce and systems in general.

As shown below, OPL sentences are readable by domain experts in the field for which the system is targeted, such as electronic commerce. These domain experts do not usually understand low-level programming languages, and even if they do, it is not realistic to expect them to read code lines. OPL is therefore used for customer verification and validation. The fact that same OPL text file is the basis for generation of the two important application facets: executable code and database schema. Changes in the analysis, design and specification are most automatically reflected in the final application.

To exemplify OPL in an electronic commerce context, consider the following example of a generic manufacturing company. The Object-Process Language (OPL) sentence set is equivalent to and extracted automatically from the OPD of Figure 1 is listed in Figure 2.

![Figure 2: Process Pre-Product Supply & Production consumes Work-In-Process Items.](image)

![Figure 3: The Object-Process Language (OPL) sentence set that is equivalent to the OPD of Figure 1](image)

![Figure 4: Zooming into the process Post-Product Supply & Production and unfolding associated objects](image)

The textual OPL and the graphic OPD specifications are equivalent to the extent that each one of them is completely recoverable from the other: given the OPD, its corresponding OPL sentence set can be derived, and vice versa.

The expressive power of the combination of the two is high due to the graphics-text synergy: a human who experiences difficulty in understanding the meaning of some symbol or interaction while examining the OPD can at any time consult the equivalent OPL text. Conversely, a person who prefers the textual OPL model can compare it to the corresponding OPD graphic model and get some insights that are not explicit in the text due to its linear nature.

5. COMPLEXITY MANAGEMENT

Object-oriented approaches usually cope with the inherent complexity of systems by applying aspect-based decomposition, i.e., they break the description of the systems into several models, each depicting another system aspect, such as structure, behavior, or function. This inevitably brings about the emergence of the model multiplicity problem, discussed above. OPM, on the other hand, handles the complexity of systems by applying detail-level-based decomposition. The idea behind detail-level-based decomposition principle is to present the system or parts thereof at various detail levels using recursive seamless scaling within OPM's single object-process model, as this paper shows.

Selective scaling provides for focusing attention on various system portions at desired detail levels without losing the big picture. A rule of thumb is to avoid diagrams that span across more than a page or contain more than about 20 entities (objects, processes and states). The OPD set comprises all the OPOs that specify the system at a variety of detail levels, from the most condensed and abstract top-level of the system OPD through intermediate detail levels to the bottom, most detailed level. The information contained in the OPD set and its equivalent OPL sentence set completely specify the system under consideration.
analogous to the Context Diagram used while analyzing a system with Data Flow Diagrams (DFDs). Other, increasingly lower-level OPDs, are scaled-up views into the details of a subset of these compound objects and processes.

![Diagram](image)

**Figure 5.** The Object-Process Language (OPL) sentence set that is equivalent to the OPD of Figure 4

Figure 4 zooms into the process Post-Product Supply & Retailing. Both the OPD and the equivalent OPL sentence set show that each one of the three processes, Transporting, Distributing, and Retailing, affect End Product. However, they do not specify how exactly the effect is manifested. To see this, in Figure 6 we express the states of End Product. This state expression is a scaling-up operation, which exposes the states of an object, thereby enabling us to explicitly specify which process is responsible for moving the object from one state to another.

![Diagram](image)

**Figure 6.** The states and state transitions of End Product

In Figure 7, which is the corresponding equivalent OPL sentence set, we can read this explicitly. The OPL sentence End Product has states manufactured, transported, distributed and relabeled, tells us what are the states of the product. The next OPL sentence, Manufacturing & Assembly yields End Product at state manufactured, tells us that the initial state of End Product is manufactured.

End Product has states manufactured, transported, distributed and relabeled.

Manufacturing & Assembly yields End Product at state manufactured.

Transporting affects End Product from manufactured to transported.

Distributing affects End Product from transported to distributed.

Retailing affects End Product from distributed to relabeled.

Consuming is enabled by End Customer, acting as agent.

Consuming consumes End Product.

**Figure 7.** The OPL sentence set of the OPD in Figure 6

What is involved in credit card processing?

The steps in credit card processing are as follows.

1. **Authorization:** The merchant must first obtain authorization for the charge from the merchant's credit card processing company. Authorization simply means that the card has not been reported stolen, and there is sufficient credit on the card. It results in the customer's credit limit being temporarily reduced by the value of the transaction.

There are two ways in which authorization may be obtained:

   1. Manual: The merchant issues a ``swipe"" of the card at the time of sale. The merchant then requests authorization using their normal method such as a point of sale (POS) terminal or PC program.
   2. Automatic: The server software communicates directly with the credit card processing company computer and arranges authorization on-line. Clearly, option 2 is preferred, but this is more complex and the costs are greater.

2. **Capture:** The final step is for the merchant to be debited. This can happen at the same time as authorization provided the merchant guarantees that delivery will take place within a certain fixed time. Otherwise capture should take place when the goods are shipped.

If the merchant's business is such that capture can take place immediately, then this can also happen automatically. Otherwise a second manual process is required.

**Charge back:** Regrettably there is sometimes a further stage where the customer is dissatisfied and attempts to persuade the merchant to cancel the transaction. Because many internet sales are made to overseas customers, many banks perceive that there is an increased risk of charge backs. It has been reported that some merchants will not accept orders from Russia because of the frequency of charge backs. Note that the fact that a payment has been authorized by the bank does not provide any protection against charge back.

**Figure 8.** Natural language and schematic description of electronic shopping and credit card processing

Then, there is a series of three processes, Transporting, Distributing, and Retailing, each of which switches End Product from one of its states to the next in its lifecycle. The OP
nows this very clearly and vividly, as are the OPL sentences. 8. Transporting affects End Product. Note that in order to void cluttering the OPD of Figure 6, only some of the details written in the OPD of Figure 4 are repeated, while most are new. Combining the two OPDs would provide a more complete picture of this portion of the system, but the resulting OPD would also be more cluttered. This distribution of details again demonstrates the principle of selective scaling and the fact that a complete understanding of the system can be obtained only by consulting all the OPDs in the OPD set of the system.

Figure 9. Zooming into the Retailing process

Figure 8, taken from the site adapted from [11], describes credit card processing in a free natural language, while an accompanying scheme provides the details of electronic shopping that precedes the credit card processing. The description in Figure 8 concerns details of the Retailing process, which appears in the OPDs of Figure 4 and Figure 6 and their corresponding OPL sentence sets in Figure 5 and Figure 7. We will therefore continue our description of the basic Retailing process by zooming into it. The result is the OPD depicted in Figure 9. The graphic specification through the OPD and the textual one through the OPL complement each other and reinforce the understandability of the system.

The structure and behavior of Retailing are so clear that there is not much to be added without simply repeating the content of the OPL sentences, perhaps in different wording. Figure 10 through Figure 13 are recursively drilled-down views of portions of their predecessors. In Figure 10 we zoom into the process Credit Card Processing of Figure 9, for which Authorization is its first sub-process. In Figure 11, we zoom into Authorizing and see that Verifying is its last sub-process, which, in turn, is detailed in Figure 12. Finally, in Figure 13 we zoom into Credit Limit Checking. At this rather detailed level, we can see explicit control structures, which are readily translatable to executable language. We have written a compiler to OPL that automatically generates C++ and Java code and applied it to several case studies of real-time systems.
OPM appeals to the human intuition as it combines system structure and behavior into a single model that caters to our natural train of thought humans apply while trying to understand a complex system. As shown throughout this paper, the synergy of combining structure and behavior into the formal graphic representation along with corresponding textual representation yields a system specification tool with high expressive power. Due to these virtues of formality on one hand and intuitiveness on the other hand, Object-Process Methodology is most suitable as infrastructure for Internet-based systems engineering environment, within which inter-company business processes and other electronic commerce activities can be defined and conducted. We should, however, reiterate that Electronic Commerce is but a case in point that demonstrates generality and applicability of OPM in the domain of system development.

REFERENCES


6. SUMMARY

This paper has presented the Object Process Methodology as a viable approach for precise and explicit specification of systems in general and electronic commerce systems in particular. To demonstrate the relevance of OPM to electronic commerce we first analyzed a broad, generic supply chain management system and its supporting electronic commerce infrastructure. Gradually, we narrowed our focus to the electronic commerce aspects of the system. In particular, we focused on the final stages of electronic commerce that take place between the retailer and the customer. Within the retailing process we further focused on credit card based payment, which is a broadly practiced electronic commerce activity. While noting that business-to-business electronic commerce can also practice this method of payment, we restricted the analysis to the interaction between the end customer and the retailer with the involvement of the credit card processing company.