The Fusion Object-Oriented Method: 
an Evaluation*

Alain Pirotte¹, Thierry van den Berghe¹, Esteban Zimanyi²

1 University of Louvain, IAG, 1 place des Doyens, 1348 Louvain-la-Neuve, Belgium, e-mail: pirotte@info.ucl.ac.be, vandenbergh@qant.ucl.ac.be
2 University of Brussels, INFODOC, 50 Av. F. Roosevelt, C.P. 175-2, 1050 Brussels, Belgium, e-mail: ezimanyi@ulb.ac.be

Abstract. This paper presents a critical evaluation, from a computer science point of view, of the Fusion method³ for object-oriented development and it sketches some guidelines for improvement. It focuses on three critical observations: (1) weakness of the ontology of the object model, (2) lack of formality, and (3) failure of the iterative process to construct comprehensive requirements and analysis models. A longer version of the paper [3] illustrates our observations with examples and develops our suggestions for improvement.

Keywords: object-oriented software development, analysis and design methods, Fusion, evaluation

1 A Very Short Introduction to Fusion

Software development remains a complex and expensive task. Although the research and development activity has been steadily growing in the field, progress has been very slow over the years.

Fusion [1] is one of several interesting recent methods for object-oriented (OO) software development (see, e.g., [2], for a collection of short descriptions). Very generally, those OO methods break system development into several stages along the system lifecycle. Further they provide guidelines along two dimensions: (1) they define representation languages or formalisms (e.g., data models), with their syntax, often graphical, and semantics to describe the structural, behavioral, and functional aspects of the system being built at various stages of its development, and (2) they propose guidelines on the process to be followed for how and when to build descriptions in the representation languages, to proceed between languages along the system lifecycle, and to perform various checks of correspondence and adequacy.

Fusion adopts the division of the development process into analysis, design, and implementation. The goal of analysis is to capture as many of the requirements on the system as possible. This is accomplished by constructing several

* This work is part of the EROOS (Evaluation and Research on Object-Oriented Strategies) project, principally based at the University of Louvain and the University of Brussels.

³ We follow the current usage, which tends to prefer “method” to “methodology”.

models of the system describing what a system does rather than how it does it.\textsuperscript{4} Specifically, the analysis models of Fusion comprise:

- an object model, that superficially resembles a usual OO database model, with object and relationship classes, and generalization and aggregation mechanisms;
- a system interface, that identifies external agents in the environment of the system, and input and output events generated, respectively, by the external agents and by the system;
- an interface model, comprising (1) an operation model, with the specification of system operations that can be invoked by input events and executed by the system, and (2) a lifecycle model, prescribing the allowed sequences of events in the form of regular expressions.

The design stage decides how to represent the system operations by the interaction of related objects and how those objects gain access to each other. Specifically, the design models comprise:

- an object interaction graph for each system operation of the interface model, that records how functionality is distributed among objects;
- visibility graphs, where communication paths between objects are detected;
- class descriptions, specifying the internal state of object classes and the external interface for methods;
- inheritance graphs, extending the generalization structure of analysis to design objects.

Implementation gives some guidelines for translating the lifecycle model into a finite-state automaton and for coding class definitions. Consistency of the various models and performance are also briefly considered.

Fusion also maintains a multi-purpose data dictionary, which turns out to play a central role (1) for cross-checking models for completeness and consistency, (2) for recording information about the development process and about the relationships between the Fusion models and the underlying real world, but also (3) for making explicit those descriptions that cannot be expressed in full because of the limited power of some models.

\section{2 Poor Ontology of the Object Model}

Many problems originate from the poor definition of the basic ingredients, objects and relationships, of analysis (or, as is sometimes said, of the ontology of the models, i.e., a precise syntactic and semantic definition of what is taken for granted, namely the languages available for expressing the structural, behavioral, and functional models). There is constant ambiguity, in Fusion and also in other OO methods, between, on the one hand, objects that reflect entities in the

\textsuperscript{4} Italics signal quotes from the Fusion reference [1].