Chapter 4.23
Dimensions of UML Diagram Use:
A Survey of Practitioners

Brian Dobing
University of Lethbridge, Canada

Jeffrey Parsons
Memorial University of Newfoundland, Canada

ABSTRACT

The UML is an industry standard for object-oriented software engineering. However, there is little empirical evidence on how UML is used. This article reports results of a survey of UML practitioners. We found differences in several dimensions of UML diagram usage on software development projects including; frequency, the purposes for which they were used, and the roles of clients/users in their creation and approval. System developers are often ignoring the "use case-driven" prescription that permeates much of the UML literature, making limited or no use of either use case diagrams or textual use case descriptions. Implications and areas requiring further investigation are discussed.

INTRODUCTION

The unified modeling language (UML) emerged in the mid-1990s through the combination of previously competing object-oriented analysis and design (OOAD) approaches (Booch, 1994; Jacobson, Christerson, Jonsson, & Overgaard, 1992; Rumbaugh, Blaha, Premerlani, Eddy et al., 1991), along with other contributions to modeling complex systems (e.g., Harel, 1987). Control over its formal evolution was placed in the hands of the Object Management Group, which recently oversaw a major revision to UML 2.0. The UML became widely accepted as the standard for OOAD soon after its introduction (Kobryn, 1999) and remains so today (Evermann & Wand, 2006). A large number of practitioner

articles and dozens of textbooks have been devoted to articulating various aspects of the language, including guidelines for using it. More recently, a substantial body of research on the UML has emerged, ranging from proposals for extending the language (Moore, 2001; Odell, Van Dyke, & Bauer, 2000) to ontological analysis of its modeling constructs (Evermann & Wand, 2001a, 2001b) to analysis of the language's complexity (Siau & Cao, 2001, 2002; Siau, Erickson, & Lee, 2005) and experiments that evaluate various aspects of the effectiveness of UML models (Burton-Jones & Weber, 2003, Burton-Jones & Meso, 2006).

The UML was not developed based on any theoretical principles regarding the constructs required for an effective and usable modeling language for analysis and design; instead, it arose from (sometimes conflicting) "best practices" in parts of the software engineering community (Booch, 1999; Booch, Rumbaugh, & Jacobson, 1999). This resulted in a language containing many modeling constructs, which has thus been criticized on the grounds that it is excessively complex (DeJong, 2006; Dori, 2002; Kobryn, 2002). But, at the same time, the UML has also been criticized for lacking the flexibility to handle certain modeling requirements in specific domains (Duddy, 2002). As a consequence, the UML has evolved to allow for the definition of "profiles" that have enabled domain specific languages (Cook, 2000; DeJong, 2006).

While the UML is intended to be "largely process-independent," some of the key originators recommend a use case-driven process (e.g., Booch et al., 1999, p.33). A majority of UML books since then have endorsed this view, and most contain at least some further prescriptions for applying the language in modeling (Larman, 2005; Schneider & Winters, 2001; Stevens & Pooley, 2000). As would be expected with a best practices approach, their prescriptions sometimes differ. While some accept the original view that only use case narratives (or, more simply, use cases) be used to verify requirements with users (Jacobson, Ericsson, & Jacobson, 1994), others explicitly or implicitly indicate that other UML diagrams can be used for this purpose, for example activity diagrams "can be safely shared with customers, even those unfamiliar with software engineering" (Schneider & Winters, 2001, p.67).

There are also differences in guidelines for using the language, and use case narratives in particular (Dobing & Parsons, 2000). This is not surprising since the official UML 2.0 documentation provides no guidance on Narrative format, stating only that "use cases are typically specified in various idiosyncratic formats such as natural language, tables, trees, etc" (Object Management Group, 2005, p.574).

Finally, when the use case-driven approach is used, concerns have been raised about the potential communication disconnect (Dobing & Parsons, 2000) that can occur when use cases are the primary communication tool among analysts and the clients or users on the project team while class diagrams play that role amoung analysts and programmers. While use case narratives have been found to be the most comprehensible artifact for managers, users and domain experts, they are the least comprehensible for designers and programmers (Arlow & Neustadt, 2004) when they require knowledge of the organizational context that programmers do not have. Conversely, class diagrams are highly comprehensible by programmers, but not clients or users (Arlow & Neustadt, 2004).

In view of these issues, it would not be surprising to find a variety of practices followed by UML practitioners. We believe understanding current practice can make an important contribution to both theoretical and applied research on UML. From a theoretical perspective, understanding how the language is used can support or challenge theoretical analyses of UML capabilities and deficiencies (Evermann & Wand, 2001a, 2001b). From a practical perspective, usage patterns can inform best practices.