Chapter 2.2
An Ontology Based Representation of Software Design Patterns

Jens Dietrich
Massey University, New Zealand

Chris Elgar
SolNet Solutions Limited, New Zealand

ABSTRACT

This chapter introduces an approach to define Design patterns using semantic Web technologies. For this purpose, a vocabulary based on the Web ontology language OWL is developed. Design patterns can be defined as RDF documents instantiating this vocabulary, and can be published as resources on standard Web servers. This facilitates the use of patterns as knowledge artefacts shared by the software engineering community. The instantiation of patterns in programs is discussed, and the design of a tool is presented that can x-ray programs for pattern instances based on their formal definitions.

INTRODUCTION

Design patterns are artefacts used to share knowledge across particular communities. Software Design patterns (Gamma, Helm, Johnson, & Vlissides, 1995) describe structural and behavioural properties of software, and are used by the software engineering community in order to exchange knowledge about software design. Currently, this is done the old-fashioned way: patterns are published in books or on Web sites, and consumed by people reading and applying them. This includes recognising pattern instances in programs in order to comprehend the structure of complex programs, and using patterns as design blueprints in order to address a specific design problem.

The main limitation with this modus operandi is the lack of tool support—patterns have to be found, and good and appropriate patterns have to be selected manually. A more effective, tool supported way of processing patterns (in particular instantiation and recognition) requires a formal representation of the patterns that supports reasoning about patterns. There are some obvious con-

tenders of modeling frameworks and languages that could be used for this purpose. This includes first order predicate logic, higher order logic, and UML based modeling languages (either profiles or separate, MOF based modeling languages). In recent years, several approaches to Design pattern formalisation have been proposed based on these choices. While these representations allow us to reason about the internal structure of a pattern, there is limited support for reasoning about the patterns themselves, their metadata, and relationships to other patterns. Moreover, logic based and UML based approaches usually lead to monolithic, static models. While this has advantages (for instance, to ensure the consistency of the models) it does not support the sharing of patterns in an open environment like the Internet. Here, knowledge is distributed and inherently inconsistent, and additional means are needed to deal with inconsistency. This is one of the key issues addressed by the Semantic Web initiative endorsed by the W3C (Berners-Lee, Hendler, & Lassila, 2001). The Semantic Web is based on the idea of a distributed, open knowledge base where everybody can publish knowledge in the form of simple subject-predicate-object assertions. This leads necessarily to inconsistencies that must be resolved. The key to solving this problem is the prioritisation of knowledge based on metadata annotations—knowledge is selected based on the author, the time of creation, and other explicit and harvested metadata.

Another issue is whether it is realistic to hope that the software engineering community will adopt one particular model or vocabulary to represent patterns. It seems more likely that there will be multiple models, each supported by a particular community, standard body, or vendor group. Mappings defining transformations between instances of the respective models will have to be developed. In the case of formal logic, this is only possible with higher order constructs describing the relationship between different predicates, functions, and types. For UML based models, standards to transform instances of different meta models are only emerging now in the realm of model driven architecture (MDA), in particular QVT ("MOF QVT Final Adopted Specification," 2005). On the other hand, modern ontology languages like OWL have built-in support to express the relationship between different ontologies.

We claim that a modeling language suitable to publish Design patterns on the Web should meet the following requirements:

1. A formal semantics is needed in order to safeguard reasoning about the patterns.
2. Pattern definitions must be easy to process by tools. In particular, this is the case if an XML based serialization format is supported.
3. There must be facilities to describe the relationships with alternative models, languages, or pattern vocabularies.
4. The physical distribution of pattern definitions and the separation of schema and instances must be supported. The pattern definition language itself should be deployed as a network resource and single pattern definitions should reference this resource. In particular, it should be possible to validate the pattern definitions against such a central schema. This is similar to the validation of XML documents against their XML schema or DTD. Furthermore, patterns refining other patterns can also refer to them as network resources, and clients can easily resolve these references using standard network clients.

It appears that a pattern deifintion language based on the Web ontology language OWL (McGuinness & Harmelen, 2004) meets these requirements. Firstly, OWL has a formal semantics (Patel-Schneider, Hayes, & Horrocks, 2004) and built-in derivation rules which support safe reasoning about patterns. This can be used to infer additional knowledge from models or to check them for consistency. Secondly, OWL ontologies