The field of Automated Software Engineering (formerly known as Knowledge Based Software Engineering) has for some years explored the use of AI and formal methods to automate parts of the software development process (including design, coding, analysis, testing, and maintenance). The goal of these efforts has been to improve productivity of the development process and the quality of the resulting software.

The semantic web brings new opportunities to this field. Web language features (hypertext and global naming) have been combined with knowledge representation language features within semantic web languages, and standardization of these languages brings with it a greater availability of tools and a broader base of knowledgeable users. These hybrid language features of OWL and RDF are particularly useful for software engineering since they bring with them inherent capabilities for distributed development and deployment. Coupled with emerging smart search technologies, these capabilities may finally address a major failed promise within software development-shared components.

Diversity is the rule in the rapidly changing field of software development. For example we have:

- methodology and model based development approaches, which are intended to produce efficient, robust, and maintainable software,
- agile/extreme programming approaches, which are lightweight and emphasize short development cycles and rapid prototyping, and
- aspect-oriented techniques, which allow the horizontal factoring of software across any number of classifiable characteristics. All of these approaches have merits and are likely to continue to be used going forward.

Methodology and model based software development approaches, in particular, have seen a move towards semantically richer styles in the specification of software systems, business processes, and the assets that such specifications are created to enable, control, and manage. Model-driven development tools and approaches already in use manage complexity, provide traceability from model to code, and provide the platform independence that enables easier migration and replication in different integration environments. This shows that model driven approaches are practical and have real benefits for software development. It’s not a big leap from this situation to using modeling languages which have precise formal semantics such as semantic web languages.

Indeed, a bridge has been built between the model driven software engineering and knowledge representation (KR) communities through the Ontology Definition Metamodel specification (ODM; http://www.omg.org/docs/ptc/06-10-11.pdf) adopted by the Object Management Group (OMG). This specification provides metamodels and UML profiles for a number of KR languages such as RDF, OWL, and ISO Common Logic. But while this bridging enables sharing and communications of models between model-driven development and semantic web environments, it is only an evolutionary step in bringing together communities that capture real world knowledge in a machine-processable way. A revolution in software development – actual automation of some of the engineering processes – will require a fusion of software development tools with the semantic web technologies for inferencing, knowledge discovery, and knowledge application. Software development tools will need to embed semantic web tools within them or embed themselves within the semantic web.

The Software Engineering Task Force of W3C’s Semantic Web Best Practices and Deployment Working Group defined a vision for this fusion of the semantic web and software engineering in “Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering.” This same group organized the 1st workshop on Semantic Web Enabled Software Engineering (SWESE) at the 4th International Semantic Web Conference (ISWC) in 2005 to discover a promising path for making this fusion occur. But the diversity of the work exploring these possibilities presented at the workshop and the visions that inspired that work was greater even than that of software development approaches. While the workshop discussion was very stimulating, no consensus was found.

This special section on Semantic Web Enabled Software Development presents a deeper look at topics covered by a few of the most interesting papers presented at the first SWESE workshop. These papers all discuss using semantic web languages to
describe some aspect of software or accompanying development activities:

- web services in Sabou and Pan [1],
- business components in Korthaus et al. [2],
- feature models in Wang et al. [3], and
- software patterns in Dietric and Elgar [4].

All but the Feature Model paper [3] propose using the web features of these languages to improve discovery and reuse to better enable sharing of software assets.

There is an interesting diversity, though, in the way machine reasoning is utilized in these papers. Dietric and Elgar [4] describes a scanner which uses reasoning to extract patterns from JAVA code. Wang et al. [3] uses the capabilities of Description Logic reasoners in the analysis of OWL DL encoded software feature models, both to check the consistency of a feature model and to test the validity of a given configuration with respect to a feature model. Korthaus et al. [2] discusses using machine reasoning for discovery and compatibility checking, although the paper focuses more on how to produce the semantic annotation that will enable this. Similarly, Sabou and Pan [1], motivated by the weaknesses they found in extant web service repositories, focus on ontology learning techniques for fleshing out ontologies describing web services. However, part of this process exploits machine reasoning to help a user find and evaluate content for semantic import.

These papers demonstrate fusion (via the embedding mentioned earlier) of semantic web languages and tools with software engineering tools, repositories, and processes. They also point to some of the areas that will need to be addressed if we are to move to semantic web enhanced software development. These areas include:

- building and managing quality semantic content to support software development,
- presenting semantic content and reasoning tool output in a form understandable to software developers,
- enforcing the semantics in this content in the code generation tools themselves, so that the generated code reflects modeled constraints, and
- using semantic content in evaluating both specifications and code for correctness, completeness, and consistency.

This special section is only a first look at some of the work exploring the possibilities of a closer fusion of semantic web with software engineering. A second SWERE workshop (SWERE2: http://km.aifb.uni-karlsruhe.de/ws/swese2006) was held at ISWC 2006, with selected papers expected to appear in another journal special section. The SWERE workshops are likely to continue in the future as a place where researchers working in this area can come together to compare their work and where the progress towards semantic web enhanced software engineering can be documented for public scrutiny and consumption.

References


Evan Wallace

Manufacturing Systems Integration Division, National Institute of Standards and Technology, Gaithersburg, MD, United States
E-mail address: ewallace@nist.gov

16 January 2007
Available online 19 January 2007