Managing Dependency Knowledge for Software Development

Steffen Staab
Fraunhofer Institute for Industrial Engineering
Nobelstr. 12, D-70569 Stuttgart, Germany
`Steffen.Staab@iao.fhg.de`

August 3, 1998

CEUR-WS.org/Vol-14/staab.pdf

The development of software requires frequent, unstructured communication and the acquisition, elaboration, administration and documentation of a huge amount of knowledge. Rather often the development process requires the participation of experts from different backgrounds: users that share their application specific background knowledge with analysts, GUI design experts who investigate the structure of the processes in which the users work everyday and experts who provide knowledge about technical facilities. All these participants in software development need to communicate their expertise - otherwise the different“puzzles may not add up to a smoothly running system.

Our interests lie in providing help to administrating the knowledge repository. This repository is used for synchronizing the experts and for identifying and distributing the knowledge which is relevant for the single developer. Two basic ideas are underlying our approach. First, we assume a common pool of knowledge about dependencies between the object model and the sequence diagrams created by the analysts and the use cases and scenarios identified by GUI designers (cf. Fowler [1]). For instance, given that a use case has to be refined in order to account for users' needs this may have effects on parts of the object model. Therefore, those developers responsible for these parts should be informed and a suggestion should be given as to how the model should be adapted.

Second, we want to exploit the semi-formal structure which is given in a typical software engineering approach in form of subconcept-of and different part-of-relations - commonly referred to in software engineering as subclass-of and feature-relations, respectively (cf. Winston, Chaffin & Herrmann [3] on part-of-relations). By this way knowledge may be efficiently organised and one's interest in a particular topic may be detected through reasoning along these relations (cf. Schulz & Romacker [2]). For instance, it may be given that a particular developer is not interested in printing procedures, in general, but in printing facilities for contracts. From subclassification and from reasoning along the part-of-relation it may be deduced that this developer is also interested in the printing procedures for life insurance contracts. Thus, she will be informed about changes that occur to this method and not only about those parts that were explicitly marked as interesting by her.

Thus, we intend to facilitate the administration and synchronization of knowledge during the acquisition and elaboration phase. However, this type of knowledge may also be used for documentary purposes after the project has been finished. Prospective users may browse the final knowledge repository with similar means such as have been provided for the developers. Users may learn about the printing of contracts with queries similar to the ones stated by the software developers.

References

[1] Fowler, Martin & Kendall Scott (1997). *UML Distilled. Applying the Standard Object Modeling Language.* Object Technology Series, Addison-Wesley.

[2] Schulz, Stefan & Martin Romacker (1998). A case study of reasoning along part-whole hierarchies in Medicine. In: *KEML'98 - Proc. 8th Workshop on Knowledge Engineering: Methods and Languages.* Karlsruhe, Germany, 21-22 January 1998 (To appear).

[3] Winston, Morton, Roger Chaffin & Douglas Herrmann(1987). A taxonomy of part-whole relationships. *Cognitive Science*, 11:417-444.