Criteria to Assess the Adaptability of Software Engineering Approaches

Katja Andresen, University of Potsdam, Germany; E-mail: Katja.Andresen@wi.uni-potsdam.de

Norbert Gronau, University of Potsdam, Germany; E-mail: Norbert.Gronau@wi.uni-potsdam.de

ABSTRACT

Because of the characteristics of contemporary system development and system evolution, it appears necessary to come up with an approach that allows the permanent and evolutionary engineering of software for new and already-existing information and application systems. The requirement is to plan and construct with room for changes, adoptions and enhancements since this will happen in the lifetime of every large software product. This contribution provides first research results to plan for adaptability in software engineering based on the research project IOSEW[1]. For that criteria are explored which promise assessing software engineering approaches for adaptability. The contribution is submitted as research in progress paper.

FOUNDATION

Adaptability applies biologically-inspired methods to enhance the capabilities of the system in focus. Adaptability comprises several qualities. Within this contribution the term adaptability is to be understood as the ability of a system to adapt itself efficiently and fast to changed demands. An adaptive system is an open system that is able to adapt its behaviour according to changes in its environment or in parts of the system itself. Additionally, it recognises the demand for change itself. The structure is determined autonomously on the basis of the systems purpose and information exchange with the environment.

VIEW ON SOFTWARE ENGINEERING APPROACHES

What would adaptability mean for software engineering approaches? The goal is that the software system can be adapted over time by humans to changing conditions (termed as design-adaptability by Oreizy and Gorlick, 1999). Transferring the cybernetic view on systems on engineering approaches, the system elements would comprise the procedure model (e.g. waterfall, v model) and phases, the (human) actors also tools and surrounding conditions as organizational units, pre-set decisions et cetera.

CRITERIA TO ASSESS SOFTWARE ENGINEERING PROCESS MODELS

This section identifies criteria to reveal the adaptive capacity of software engineering procedures. The first set originates from factory planning but has also been successfully applied to assess and design for adaptability in information systems [Andr06]

Scalability

Scalability refers to the permanent state to operate effectively and efficiently at many different scales. A system is supposed to be scalable if it will remain effective when there is a significant increase/decrease in the number of recourses.

For the software development process scalability stands for the seamless addition or reduction of system elements as actors, resources, tools, functions. The process model is scalable if control structures as loops and iterations can be inserted or deleted leading to the above mentioned steps. The evaluation might be based on diagnosis features.

Modularity

Modularity generally means the structuring of a system into small, partly autonomous subsystems [WiPi+97]. A module consists of a module trunk and a module interface. The interface contains a specification on the characteristics to connect and communicate, which are of importance for its surrounding field.

The phases of a procedure model correspond to the idea of modules. Each phase defines a function, e.g. requirements analysis or testing. Within each phase activities and results may be defined. For example, the result of the requirements phase as a set of documentation.

Interoperability

This indicator refers to the ability of resources to place a high measure of compatibility and connectivity. It is commonly realised by deploying interface standards in the domain of information system architectures [HaSt03].

Interoperability in the context of procedure models stands for the transfer of input or output from one phase to another. The results of the requirements phase should be applicable in the next phase.

Independence

Independence raises the question of spatial and temporal unlimited access to subsystems (components). In information systems the criteria independence is interpreted as unlimited access to applications, functions and data by different technologies as web-browser, terminal-server for instance. Within software engineering the criteria adds an organizational aspect. If the phases of the procedure model are independent they can be performed at any location, in any country at any time for instance.

Redundancy

It is a principle in nature that important functions can be taken over by a near-by system if another breaks [Vest02]. Redundancy requires a fall-back strategy, if failure or unforeseen events happen. An engineering approach considering redundancy checks for results by control strategies as feedback loops and provides alternatives to solve process steps. Does the procedure model provide support for the change of organizational units along the development process?

Self-Organization

Self-organization (autopoiesis) marks the ability of a system to determine the systems structure by adjusting and steering mechanisms related to processes within the system in order to ensure the long-term existence of the system [MaVa87]. Thereby the resources and subsystems produce their own order by taking up information about their environment and their reciprocal effect with the environment.

Self-organization in software engineering is a goal oriented process whereby the phases define the main goals. However, the engineering process model does have to ensure the correct result along with the adequate means to (efficiently) deliver the outcome. It requires learning qualities as double-loop learning. Self-organization is one of highest form of behaviour of complex systems.

Knowledge

Knowledge about the engineering model denotes both – the person-bound and the explicit knowledge [GrKo+05]. The latter is presented for example in the form of writing fixed processes, rules, guidelines, responsibilities, communication channels, programming language and more.

Knowledge and information access support the rapid implementation of functional and organisational changes in general or the handling of the software specifically.

NEXT STEPS

A logical next step is the application of the criteria to discuss established process models as the V model, RUP (Rational Unified Process or eXtreme programming).

REFERENCES

[Andr06] Andresen, K.: Design and Use Patterns of Adaptability in Enterprise Systems, Gito, 2006.

[HaSt03] Hanisch, F., Strasser, W.: Adaptability and interoperability in the field of highly interactive web-based courseware. In: Computers & Graphics, Volume 27, Issue 4, August 2003; 647-655.

[GrKo+05] Gronau, N.; Korf, R.; Müller, C.: KMDL - Capturing, Analysing and Improving Knowledge-Intensive Business Processes. Journal of Computer Science, 4, 2005; 452-472.

[OrGo99] ‚Oreizy, P., M. M. Gorlick, et al. (1999). "An Architecture-Based Approach to Self-Adaptive Software."IEEE Intelligent Systems 14(3): 54-62.

[Vest02] Vester, F.: Vester, F.: Die Kunst vernetzt zu denken: Ideen und Werkzeuge für einen neuen Umgang mit Komplexität -der neue Bericht an den Club of Rome. Deutscher Taschenbuch Verlag, 2002.

[WiPi+97] [WiPi+97] Wiegand, R. T., Picot, A., Reichwald, R.: Information, Organization, Management, Wiley, 1997

[MaVa87] Maturana, H., Varela, F.: The Tree of Knowledge. Shambhala Publications, Boston, 1987;

[High00] Highsmith, J. A.: Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. New York, Dorset House Publishing.

ENDNOTE

[1] IOSEW: Project on Intra-Organizational Software Engineering and Adaptability funded by German Research Ministriy (http://www.iosew-projekt.de)