Chapter 2.12
Knowledge Patterns

Jörg Rech
Fraunhofer Institute for Experimental Software Engineering, Germany

Raimund L. Feldmann
Fraunhofer Center for Experimental Software Engineering, USA

Eric Ras
Fraunhofer Institute for Experimental Software Engineering, Germany

INTRODUCTION

Knowledge patterns are one way to formalize and describe lessons learned and best practices (i.e., proven experiences) about structuring knowledge, the design of KM systems, or the development of underlying ontologies. Such patterns capture aspects that positively or negatively influence the KM activities. In the later case, where negative influences are described, such patterns are denoted as anti-patterns. Knowledge patterns and anti patterns support practitioners and researchers in their knowledge management (KM) activities and can help in developing KM systems as well as improving the quality of the systems themselves and that of the knowledge within (i.e., the quality of the knowledge). Thereby, patterns in KM represent a way of structuring knowledge as well as a form of language that helps knowledge engineers to communicate about knowledge and KM systems.

DOI: 10.4018/978-1-60960-783-8.ch2.12

BACKGROUND

Knowledge is one of the most important assets for any kind of organization, and for all areas of science. While *experiences* describe events in one specific context that can only be reused carefully, *knowledge* is usually applicable in previously unknown contexts with a fair amount of certainty. Unfortunately, a small number of experts who have acquired knowledge through their experiences in day-to-day work hold major parts of the knowledge in an organization. Surprisingly, this is equally true for researchers in KM. Experiences gained regarding knowledge itself and KM systems, either technical, social,

or socio-technical ones, are typically recorded in the form of models or process models only. Fine-grained knowledge about the structuring, interconnection, or classification of knowledge is rarely documented, and common and recurring patterns are hardly available. Further, while best practices regarding the technical KM system or KM initiatives are often documented and shared (Davenport & Probst, 2000; Mertins et al., 2003), knowledge and best practices are often hard to transfer (Szulanski, 1996).

Such knowledge about KM systems is documented in the form of success factors (Mathi, 2004) (Thomas, 2006) (Morisio et al., 2002), success models (Jennex & Olfman, 2004, 2006), success measures (Jen & Yu, 2006), reference architectures for KM systems (Davenport & Probst, 2000; Mertins et al., 2003), worst practices (Fahey & Prusak, 1998), barriers (Eberle, 2003), facilitators (Damodaran & Olphert, 2000), and incentives (Feurstein et al., 2001), which are often described in an unstructured and informal way. They typically preserve knowledge about a whole KM system or initiative. Barriers, facilitators, or incentives represent types of patterns that describe common and recurring incidents, practices, or behavioral structures in KM. There are many different types of barriers, such as knowledge barriers in general (Riege, 2005), barriers in knowledge transfer (Sun & Scott, 2005) and distribution (Bick et al., 2003), barriers based on culture (Wolf & Wunram, 2003), as well as barriers based on roles and activities (Awazu, 2004).

In software reuse, several barriers were described by Judicibus and classified into the two classes "individual factors" and "collective factors" (Judicibus, 1996), such as the "Feudal Lord's Syndrome" or the "Egghead's Syndrome" (Rech et al., 2007a).

In software engineering, design patterns are a relatively new concept, which was transferred from architecture to represent typical and recurring patterns of good and bad software architectures. These design patterns (Gamma et al., 1994) and anti-patterns (Brown et al., 1998) were the starting point for the description of many patterns in diverse software phases and products. Today, we have thousands of patterns (Rising, 2000) for topics such as software reuse (Long, 2001), agile software projects (Andrea et al., 2002) or pedagogical science (http://www.pedagogicalpatterns.org/) (Abreu, 1997; Fincher & Utting, 2002). Many other patterns are stored in pattern repositories such as the Portland pattern repository (PPR, 2005) or the Hillside pattern library (HPL, 2005) and are continuously expanded by conferences such as PLOP (Pattern Languages of Programming; see http://hillside.net/conferences/).

While there are similar concepts, the idea of patterns is relatively new in KM. Nevertheless, the concept of patterns and anti-patterns helps in documenting knowledge and experiences.

PATTERNS AND ANTI-PATTERNS IN KNOWLEDGE MANAGEMENT

The quality of the knowledge gained, the technical KM system used, or the social KM method applied is neither easy to be evaluated, nor is it easy to be improved. This is partly due to the fact, that there exists no universal KM system, which is suitable for all kinds of organizations, or universal knowledge, which is suitable in all situations. In practice, each system, as well as the knowledge within, has to be adapted and tailored to the individual needs of an organization and the people within.

In order to tailor knowledge and KM systems to the specific needs at hand, we can resort to the concept of software design patterns, which are used to structure and adapt software systems to the needs of the customers. By transferring the concept of software design patterns to knowledge management, we obtain the following definition for knowledge and knowledge management patterns: