TOWARDS PATTERN MANAGEMENT SYSTEM

Erki Eessaar
Institute of Informatics, Tallinn Technical University, Raja 15,12618 Tallinn, Estonia
Email: Erki.Eessaar@mail.ee

Keywords: knowledge management, reuse, patterns, Pattern Management System, reuse repository.

Abstract: Patterns allow division of knowledge into manageable pieces. It is necessary to develop Pattern Management System (PMS) in order to effectively use patterns. In this article components of PMS are proposed. PMS uses metadata about patterns and database of patterns to guide management of patterns. Textual descriptions of patterns and models what specify pattern are stored in the database of patterns.

1 INTRODUCTION

Reuse is a practice that uses existing knowledge to solve problems. Patterns permit division of knowledge into manageable and reusable pieces. Knowledge about the problem, its context and solution can be written down as a pattern.

Researchers have proposed repositories of reusable software components (Gaedke & Rehse, 2000), (Sugumuran & Storey, 2003). Authors like (Halaris & Geroupoulus, 1996) and (Backlund & Jonasson, 2002) have stressed that patterns can be used during all the phases of Information System Development (ISD) Process and therefore Pattern repository is needed.

Author's goal is to develop Pattern Management System (PMS), which would allow creation / storage / modification / classification / search of different types of patterns. It could be used for example during ISD. PMS could also assist development of methods for ISD (Method Engineering). According to (Baskerville & Stage, 2001), information systems development practitioners invent, combine and adapt fragments of methods to a development situation rather than wholly adopt published methods. It is a process that is similar to the usage of patterns.

PMS must support current and evolving standards, for example Pattern and Component Markup Language (PCML), which describes among other things metamodel of patterns.

The rest of the paper is organised as follows. Section 2 introduces architecture of PMS. It mostly concentrates on the structure of database. Section 3 summarizes and points to the future work with current topic.

2 ARCHITECTURE OF PMS

(Marco, 2000) describes meta-data management systems (MDMS) what help to collect and use information about the structure of information systems, processes that occur in the information systems and parties that use these systems. (Marco, 2000) predicts that bidirectional metadata architecture will evolve in the future where "meta data is changed in the meta data repository and fed back into the meta data's original source." (Marco, 2000, p. 311) Author of current article suggests that PMS should be one additional component of MDMS. For example logical and physical data models can be created using the assistance of PMS and then changes can be automatically made in the databases based on them.

This article proposes architecture of Pattern Management System (PMS) what uses relational database to store patterns data. PMS should have web-based interface so that developers around the world could use this.

2.1 Description of patterns

Description of patterns consists of two equally important portions:

- Structured textual descriptions. For example description of the problem, context where the problem occurs and solution to the problem.

– Models that complement textual descriptions, describing the problem, solutions to the problem and also examples of pattern usage.

Models present more structured view to the domain than descriptions in natural language and therefore they provide better base for searching patterns. Models can be created for example using the CASE tools what have capabilities to save models as XMI files. Information from these files could be loaded to the database of PMS. For example (Hay, 1996) describes universal data models, which can be starting point for the development of logical data models and are actually patterns. It is possible to search model-fragments using database queries if information about entities, relationships and attributes is stored in a database. If model element is found, then it is possible to search other model-fragments and their associated documentation, what contain the same or semantically similar element. For example if user decides that database must contain information about persons and organizations then we can find other patterns what reference to the entity "party" or its subtypes. It helps to find new requirements to the database.

2.2 Main components of PMS

Architecture of PMS consists of following basic layers:

– Meta layer, which allows management of meta-data about patterns. It will be used to guide queries and changes in the database, because this layer manages information about the structure elements of patterns, pattern types and corresponding database objects.
– Pattern layer, which enables management of data of actual patterns and objects what are associated with them.

Pattern management will be performed using the meta-data about patterns. For example user must select pattern type for the registration of pattern. Program will identify its structure elements (name, problem, context etc.) based on selected type and will generate user interface for the registration of pattern. Program saves data about patterns to the relational database based on the mapping between pattern structure elements and corresponding relations and attributes in the database. Next the main functional subsystems of PMS will be presented.

2.2.1 Main functional subsystems of PMS

– Subsystem for managing meta-data of patterns. It permits definition of the structure elements of pattern description styles and construction of the styles using predefined elements.
– Subsystem for pattern management. Its main functions are creation, modification, search and evaluation of patterns and pattern languages. It must also be capable to extract and load model information from XMI files to the database.
– Subsystem for managing opinions of users about patterns.
– Supporting information management subsystems for managing information about objects with which patterns have associations.

2.2.2 Sub-components of PMS database

Next the sub-components of PMS database will be presented.

1. Registers of patterns meta-data

Each pattern has description style. Logical structure elements and domains are the building blocks, which are used for the construction of the description styles.

– Register of logical structure elements of pattern description styles where data about structure elements and their attributes is registered. Structure elements are for example "context", "forces", "problem", "solution", "positive consequences of pattern usage", "negative consequences of pattern usage", "examples". Attributes could be for example "description" and "summary".
– Register of domains. Author suggests that each attribute has domain in the context of pattern description style. Domain describes the rules that apply to this attribute in the context of the style:

a) Data type of the attribute.
b) Is the value of the attribute required?
c) Is the value of the attribute unique among other patterns? For example name of the pattern should be unique.
d) What is the default value of this attribute?
e) Other constraints to the values of the attribute.

– Register of description styles of patterns. Each style consists of set of logical structure elements. Appropriate domain from register of domains is associated with each attribute of logical structure element. System must also permit creation of new styles based on existing styles. One important fact (multiplicity of element) what is registered about association of structure element and style, determines amount of values what can correspond to this structure element in the real pattern. For example each pattern can have one base name but many alias names, forces or code samples.

- Register of metamodels, which describe elements of the model types. For example class diagrams and collaboration diagrams illustrate and explain patterns of object-oriented design. Metamodels determine the structure of database where models will be stored.
- Register of physical structure of patterns database. It contains information about relations and their attributes in the database where components of patterns are kept. Central relation in the database of patterns is called "Patterns". It contains one tuple for each pattern, which is stored in the database of patterns.
- Register of pattern types. Pattern type is associated with description style and one or more domains where patterns can be used.
- Register of mappings between elements of pattern description styles and physical structure elements of database of patterns.

Addition or modification of a domain or style causes PMS to change the structure of database and record this change in the register of physical structure of patterns database and in the register of mappings. If new domain is defined then system creates domain object in the database. If structure element is added to the description style, then PMS creates new relation in the database. It will have foreign key attribute which possible values are identifiers of patterns from the relation "Patterns". Database might contain more than one relation where attribute values of same type of structure element are registered. Attributes in these relations have different domains. Each relation has associated triggers what enforce appropriate multiplicity of the element. Meta-data guides management of data in these relations.

2. Registers that contain actual patterns and user opinions about them

- Register of textual descriptions of patterns. Textual descriptions are organized according to the logical structure of pattern description style.
- Register of models what are associated with patterns. PMS has more than one such register - one for each model type. Each model type has metamodel, which describes model elements that can be used to create model of that type. Structure of each register is developed based on metamodel.
- Register of pattern languages. Pattern language is a special case of pattern what describes solutions to the set of problems. One pattern could belong to many pattern languages and language contains more than one pattern. Pattern language as pattern might have its own structure elements (name, problem etc.) and associated models.
- Register of user evaluations of patterns. Patterns can be searched based on them. Patterns have different type of descriptive attributes:

a) Attributes, which are common to all the patterns and have a same set of possible values in all domains (type A).
b) Attributes, which are common to all the patterns, but in different domains have different set of possible values (type B).
c) Attributes, which are used only with patterns from specific domains and have a set of possible values (type C).

Value from a pre-agreed structured set of terms may be associated with such attributes. Sometime it is difficult to determine exact value of attribute from the list of values. For example nowadays there is no governing organization, which standardizes patterns and decides that one pattern is still a candidate pattern but another is mature and accepted pattern. Some of such attributes (they have type A) are:

1. Success of a pattern.
2. Maturity of a pattern.
3. Usability of documentation of a pattern.
4. Level of abstraction of the proposed solution.

Creation of patterns is collaborative work. Therefore judgements of users could be used to determine the values of attributes as well.

Users could compare patterns in pairs in regard to selected attribute, using values from the fundamental scale of decision method "Analytic Hierarchies Process (AHP)". For example result of a comparison might be that pattern p_1 has essentially better usability than p_2. If new pattern is created, then the patterns manager, who has obligations to encourage the use of patterns, compares it with patterns which are well known to all or most developers for their good or bad properties. Pattern users can compare patterns what they have used.

If patterns are searched based on attributes which values have been determined by users, then it is necessary to determine other pattern, compared to which the value of an attribute is found. For example query could be: "Find patterns of type t which have better usability than pattern p_1."

Next example presents some attributes of type B and some of their possible values in the Information System Development domain.

1. Level of technicality of a problem (sample values: "Business Environment"; "Enterprise Model", "Technical model").
2. View to the system (sample values: "Data"; "Functions"; "Time"; "People").
3. Extent of the problem what pattern solves (sample values: "Subsystem"; "Component").
4. Activities what the pattern helps to perform (sample values: "Modeling"; "Programming").

Users of PMS could associate patterns with attribute (type B, C) values together with membership function value (between 0 and 1). It permits calculation of average membership function value for each pair of attribute value and pattern.

3. Registers, containing supporting information

These registers are part of any major information system. Common patterns what describe their structure have been worked out and have been published as patterns (Hay, 1996), (Fowler, 1997).

- Register of documents what contain information about patterns or what have been used to create or complement patterns. Patterns of documents data model (Hay, 1996, p. 205-234) can be used to develop the structure of this register.
- Register of parties (persons and organizations) who have some role in the events, which have happened or have been scheduled to happen with patterns. Patterns of domain models about parties (Fowler, 1997, p. 24-27) can be used to develop the structure of this register.
- Register of planned and happened events, which have happened with patterns (for example creation, review or modification of a pattern). Patterns of proposed and implemented actions (Fowler, 1997, p. 158-160) can be used to develop the structure of this register. For each event next data will be stored: time of an event; reference to the parties who participated in the event, together with the information about their roles in the event; references to the documents what where used during the event, together with the information about their roles in the event.
- Register of queries, what users of PMS have constructed and stored for later usage by them or other interested parties.

3 CONCLUSION AND FUTURE

In this paper, principles of Pattern Management System (PMS) were presented. PMS could help find solutions to the problems and store evolving knowledge in the form of patterns. Author proposes, that meta-data about structure of patterns and database of patterns must guide the operation of PMS in order to manage different types of patterns. PMS must manage both narratives and information extracted from models. It makes possible to use query capabilities of existing database management systems to search patterns based on model elements.

Future work will involve the development of methods for searching patterns. Search based on user evaluations should take into account the amount of evaluations and quality of experts who make them. One expert could evaluate patterns in regard to some attributes many times. If lot of experts have changed their opinion recently in one direction, then it could also be taken into account.

One promising methods is to search patterns according to semantic similarity. For example during requirements analysis functional requirements to the system are documented using use cases. Data model can be constructed by searching data modeling pattern which model elements (entities, attributes and associations) have the biggest semantic similarity to subset of the use case text.

REFERENCES

Backlund, P., & Jonasson, I., 2002. 'Patterns as a Means for Managing Knowledge in the Information Systems Engineering Process', in *BalticDB&IS 2002,* vol. 2, eds H. M. Haav & A. Kalja, Tallinn Technical University, Tallinn, pp. 15-25.

Baskerville, R., & Stage, J., 2001. 'Accommodating emergent work practices: etnographic choice of method fragments', in *Realigning Research and Practice in Information Systems Development: The Social and Organizational Perspective*, eds N. L. Russo, B. Fitzgerald & J. I. DeGross, Kluwer Academic Publishers, pp. 11-28.

Fowler, M., 1997. *Analysis Patterns: Reusable Object Models*, Addison Wesley, Calif.

Gaedke, M., & Rehse, J., 2000. 'Supporting compositional reuse in component-based Web engineering', *Proceedings of the 2000 ACM symposium on Applied computing*. Retrieved 17 Oct. 2003 from ACM digital library.

Halaris, J., & Geroupoulus, S.T., 1996. 'Reuse Concepts and a Reuse Support Repository', *IEEE Symposium and Workshop on Engineering of Computer Based Systems,* pp. 27-34. Retrieved 18 Aug. 2003 from IEEE Xplore.

Hay, D. C., 1996. *Data model patterns: conventions of thought*, Dorset House Pub, New York.

Marco, D., 2000. *Building and Managing the Meta Data Repository: A Full Lifecycle Guide*, John Wiley & Sons, USA.

Sugumuran, V., & Storey, V.C., 2003. 'A semantic-based approach to component retrieval', *ACM SIGMIS Database*, vol. 34, issue 3, pp.8-24. Retrieved 17 Oct. 2003 from ACM digital library.

Pattern and Component Markup Language. Draft 3. Retrieved June 19 2003 from http://www.objectventure.net/files/docs/PCMLSpecification.pdf