Reuse and Adaptation of Software Process using Similarity Measurement

Viviane Santos, Mariela Cortés and Márcia Brasil

Universidade Estadual do Ceará
Av. Paranjana, 1700 - Cep 60.740-000, Fortaleza, CE, Brazil

Abstract. Software process reuse involves different aspects of the knowledge obtained from generic process models and previous successful projects. The benefit of reuse is reached by the definition of an effective and systematic process to specify, produce, classify, retrieve and adapt software artifacts for utilization in another context. In this work we present a formal approach for software process reuse to assist the definition and adaptation of the organization's standard process. The Case-Based Reasoning technology is used to manage the collective knowledge of the organization.

1 Introduction

Considering the forward dependency between the development process quality and the product quality, the deep knowledge of the activities involved in the process and their management are critical factors for the organizational success.

In high level, the software development process defines a formal sequence of activities related to a set of artifacts, people, resources, organizational structures and constraints for turning user requirements into software. This knowledge captures the guidelines to drive software development in a specific domain and/or context.

The definition of a process for software development is a complex task since it requires experience and combines the knowledge of diverse technological and social aspects. The utilization of standards for the process definition [1][2][3][4][5] is recommended in norms, processes and maturity models. However, the process model must be adapted to fit the organization characteristics.

Software process models describe the organization knowledge and, thus, models that enhance successful experiences must be disseminated and recommended for reutilization across the organization [2][6]. The process consolidation is achieved through the systematic reuse and the incremental capture of feedback, looking for the continue improving.

The purpose of the process reuse technology is to support the process definition and improving on the basis of standard processes, according to norms and quality models, and learned experiences [7]. Dynamic and context-depending aspects of the knowledge in software development turn the Case-Based Reasoning approach (CBR) [8] useful as it provides a broad support for the dynamic management of the organiza-

Santos V., Cortés M. and Brasil M. (2009).
Reuse and Adaptation of Software Process using Similarity Measurement.
In *Proceedings of the 4th International Conference on Evaluation of Novel Approaches to Software Engineering - Evaluation of Novel Approaches to Software Engineering*, pages 215-226
DOI: 10.5220/0001954102150226

tional knowledge and continuous incremental learning necessary for the definition and improving of software development.

In this work we describe an approach for definition and reuse of the organizational standard process, on the basis of models, standards, quality norms, and previous experience, in accordance with the organizational reality and characteristics. In addition, on the basis of the reuse process results, an adaptation process is presented. The CBR technology is used for the management of the repository and the retrieval of assets.

This work is organized as follows: in Section 2 the CBR technology is briefly explained. In Section 3 the process reuse using CBR is presented. In Section 4 a case study is illustrated. Finally, final considerations are presented.

2 Case-based Reasoning

The CBR technology solves problems in a specific situation, through previous similar situations [9]. A case comprises a pair problem that describes the context of an actual case occurrence, and solution that presents the problem solution. Past cases are used to hint strategies to solve new similar problems [10].

A CBR system is composed by 4 basic elements [8]: knowledge representation, similarity measure, adaptation and learning.

The knowledge representation consists on the description of the relevant information for the cases, in order to assess the reuse.

The similarity measure establishes the global similarity degree between a base case and a new problem under consulting. This measure is based on a heuristic method [9]. The retrieval process results in a set of ranked cases that are based on the global similarity measure.

The utility of base case to solve a problem is proportionally related to the effort required to adapt it to fit the specific context [10]. This process involves knowledge reuse in problems solutions along the knowledge transference from the previous case to the actual case.

The ability to learn from early experiences is inherent in a CBR system. The continuous learning contributes to increase the system capacity to improve their interpretations to solve new problems. In this sense, feedback about the soundness and effectiveness about their interpretations is required.

3 Process Reuse Approach

The approach for process reuse is presented in Fig. 1 [11]. The main component is the Processes Assets Repository which is designed to store assets models for reuse and their attribute-value representations. This representation involves a set of relevant properties to describe each case, and the values for these properties including numeric, text, pre-defined terms, etc. The utility of a specific case from the repository in the context of a new case under consulting is enabled using this representation.

Considering that process models are abstract, their inclusion in the repository requires the existence of an instance in a specific case. The Search Engine uses CBR

technology to retrieve similar cases through the similarity measurement on the basis of process and project features. Attribute-value representations must be defined for the new case, and for the base cases in the repository.

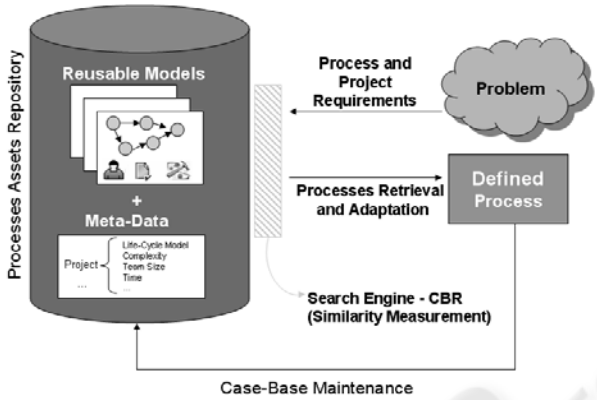

Fig. 1. Approach for process reuse.

The reutilization involves the adaptation of a previous solution for a similar case, using an appropriate method [10]. After its adaptation and execution in the new project, the reused process instance is evaluated in order to examine their effectiveness and capture reuse information. Then, the new instance of the model can be included into the repository, increasing their attribute-value representation.

3.1 Representation of Organizational Assets in the Repository

The reutilization of cases is enabled whenever the cases will be indexed and stored appropriately in the repository of process assets, in such a way to make possible its efficient retrieval. The suitable representation of the process assets is a critical factor for the success of the method, since the similarity degree for the correct retrieval of the cases is measured on the basis of this representation. The similarity concept consists of establishing an estimate of the utility of a previous case stored in the repository, in the context of the current case on the basis of the observed similarity among the representations of both cases [8].

The similarity types are restrictions applied to the representation features, to establish its correspondence or co-occurrence among cases [12]. The similarity types used in this work are:

- Numeric (NUM). Positive integer or real numbers
- Qualitative for Fixed Items (QFI). Predefined Terms
- Qualitative for Variable Items (QVI). Registered terms with possibility of new items

The similarity between cases is based on the comparison of the features in the representation and the corresponding values. In this sense, several studies related to the classification of the process assets for reuse in other contexts can be cited

[7][12][13][14][15]. The representation of the assets in the repository proposed in this work is presented in Table 1. The features had been organized in agreement to the target in process and project features.

Table 1. Representation of the assets in the repository.

Scope	j	Feature	Description	Similarity Type	Constraints
Project	1	Life-Cycle Model	Project life-cycle model, such as Cascade, Iterative Incremental, Evolutionary, Spiral.	QVI	
	2	Complexity	Project complexity: High (including critical and advanced functionalities), Medium (including feasible functionalities), Low (including simple functionalities).	QFI	
	3	Size	Project size regarding the functionalities quantity: Large, Medium or Small.	QFI	
	4	Team Size	Project integrant number.	NUM	
	5	Time	Project duration in months.	NUM	3, 4
	6	Software Engineering Knowledge	Knowledge level in Software Engineering: High (theory e practical), Medium (theory only), Low (none knowledge).	QFI	
	7	Development Paradigm	Project development paradigm, such as Structured, Object Oriented, etc.).	QVI	
Process	8	Development Model	Software development models, like RUP, XP, SCRUM, etc.	QVI	
	9	Maturity Model	Maturity model, for example, CMMI, MPS.BR, etc.	QVI	
	10	Maturity Level	Specific maturity level related to the maturity model specified previously. It can be, for example, 1 to 5 (CMMI and ISO/IEC 15504) or G to A (MPS.BR).	QVI	9
	11	Complexity	Process complexity based on the maturity levels: High (advanced levels), Medium (intermediary levels), Low (low levels).	QFI	8, 9, 10
	12	Process	Specific processes, such as Requirements Management, Project Planning, Quality Assurance, Configuration Management.	QVI	
	13	Experience on Process Usage	Team's experience on software process usage: High (process used in more than 15 projects), Medium (process used in a range of 5 to 14 projects), Low (0 to 4 projects).	QFI	6
	14	Success Level	This result (1 to 10) represents an indicator of the degree of organizational satisfaction about the adopted process.	NUM	

3.2 Retrieval Process

The most appropriate solution for the current problem is retrieved from the repository through similarity measurement. The greatest value in this measurement indicates greater similarity between the cases.

In CBR, several techniques can be applied for data retrieval. In [9] the algorithm to calculate the similarity is based on k-NN technique, where the global similarity

(SIM) between two cases (*a* and *b*) is defined by the weighted sum of the local similarities (sim_j) for each feature (A_j).

$$SIM(a,b) = \sum_{j=1}^{n} w_j \times sim_j(A_j(a), A_j(b)) \quad (1)$$

The weight (w_j) reflects the relevance of a feature (A_j) concerning the similarity of cases. This factor is determined by the user and is measured by the values: High (100), Medium (50) and Low (10). The features considered more important for the problem resolution from the user's viewpoint, possess higher weights.

The base cases with greater similarity measurements are considered as sufficiently similar and proposed to the user as reuse candidates. Note that if the same weight is assigned to all the attributes, the base case that attends the greater number of features must be the suggested one.

The local similarity is calculated in accordance with the similarity type of each feature (Table 2) and considers the computation of distance (d_j) between each feature values in the cases *a* and *b*:

$$sim_j = \frac{1}{1 + d_j(a,b)} \quad (2)$$

This measurement must be normalized [16] to avoid over influence of a metric by the great range of values of the attributes. The normalization process uses smallest and greatest values in the repository to linearly produce values between 0 and 1.

The distance between two features of numeric similarity type (NUM) is calculated on the basis of a proportionality relation between the values. Thus, the local similarity in this case is expressed as:

$$d_j(a,b) = \left| \left(\frac{A_j(a) - \min(A_j)}{\max(A_j) - \min(A_j)} \right) - \left(\frac{A_j(b) - \min(A_j)}{\max(A_j) - \min(A_j)} \right) \right| \quad (3)$$

For the Qualitative for Fixed Items (QFI) the distance is calculated by establishing a proportion between values through the fixed items: High/Large (9), Medium (6) and Low/Small (3). The expression for the local similarity for QFI features is:

$$d_j(a,b) = \left| \left(\frac{A_j(a) - 3}{9 - 3} \right) - \left(\frac{A_j(b) - 3}{9 - 3} \right) \right| \quad (4)$$

Thus, the expression can be resumed to:

$$d_j(a,b) = \left| \left(\frac{A_j(a) - 3}{6} \right) - \left(\frac{A_j(b) - 3}{6} \right) \right| \quad (5)$$

Finally, to calculate the distance between features of Qualitative for Variable Items (QVI) is used a taxonomy to hierarchically represent the relationships among the terms (Fig. A1), where *s* is the distance (jumps) between $A_j(a)$ and $A_j(b)$ in the taxonomy. The measurement for a new case may require the inclusion of new terms.

$$d_j(a,b) = \frac{s}{10} \tag{6}$$

The measurement for a new case may require the inclusion of new terms.

3.3 Adaptation Process

Adaptation involves the process to transform the retrieved results into an appropriated solution for the currently problem. The adaptation process can be realized following different approaches [9]. In this sense two approaches can be suggested: if the similarity measurement of the retrieved process in the top of the ranking is satisfactory,[1] a minimal or null adaptation can be required. In other case, when none of the retrieved processes fulfills the requirements for the new case in appropriate manner, a compositional approach is proposed.

	Local Similarity Values between each attribute of the searched cases and the attributes of the current case (the new case).				**Attribute Values to the New Case**
Attributes	**Base-Case 1**	**Base-Case 2**	**...**	**Base-Case M**	**New Case**
Attribute 1	***SimL***(A_{11})	***SimL***(A_{12})	...	***SimL***(A_{1M})	V_1
Attribute 2	***SimL***(A_{21})	***SimL***(A_{22})	...	***SimL***(A_{2M})	V_2
...	...	...	...	...	...
Attribute N	***SimL***(A_{N1})	***SimL***(A_{N2})	...	***SimL***(A_{NM})	V_N

Fig. 2. Similarity values of Base-Cases relative to the attribute values of the current case.

In this approach [17], the solution is composed by elements from different processes, on the basis of the most similar process models returned from the previous step. In this sense, the maximization of the local similarities of each feature from different models can be used to build a new case matching the greatest level of similarity to meet the new process features, considering the dependencies and constraints among the features. Fig. 2 presents, in general way, the returned cases with its features and similarity values against the new case.

Thus, the maximized global similarity of the new case (called *GlobalSIM*) is calculated through the maximization of the local similarity (*LSim*) of each feature from the retrieved cases, as presented below:

$$GlobalSIM = \sum_{i=1}^{N} \max\left(Lsim(Ai_1), Lsim(Ai_2), Lsim(Ai_3), ..., Lsim(Ai_M)\right) \tag{7}$$

[1] The satisfactory level is determined by the average of the base-case local similarities percent to represent the adherence of a case-base against the current case. The user can restrict the ranking result through specifying a minimum percent of satisfactory level, e.g. 60%.

Where *N* represents the quantity of features and *M* the quantity of retrieved cases. In addition, already dependencies and restrictions between features from the same process must be considered in the composition of the new process. Similarly, features from different process can be incompatibles. These restrictions must be considered in the composition process. In this case, the following dependencies and constraints were identified:

- Development Model and Maturity Model;
- Maturity Model and Maturity Level;

For example, if the Maturity Model feature value required is SW-CMM or CMMI, the Maturity Level feature must be values from 1 to 5. Similarly, if the required Development Model is XP, neither Maturity Model nor Level Maturity can be used. In this sense, a recently published report of the Software Engineering Institute [18] considers the possibility of joint the use of agile development methods and CMMI best practices as a way to improve the performance.

```
ALGORITHM GLOBAL_SIMILARITY_MAXIMIZE

Variables:
Vector LOCAL_SIM [2,14]: integer;

//First line: stores the case identifier where the attribute with maximum local similarity was found.
//Second line: stores the value of the greatest similarity found for the corresponding attribute to the vector index
//The number of columns (14) is the total quantity of attributes to be evaluated.

Begin
        For each attribute of the scopes Process and Project do:
                Check in all cases recovered the local similarity value of the attribute;
                Identify the case that has the greatest local similarity value for this attribute;
                Recover the attribute value;
                Check if there is conflict in the attribute values;
                While there is conflict:
                        Check the next case that has the higher value of local similarity to the attribute;
                Store in SIM_LOCAL the identifier of the case and the value of local similarity of this case;
        Realizing the sum of all maximum local similarities obtained in each attribute;
End
```

Fig. 3. Algorithm to maximize the global similarity.

The selection of the features to compose the new process involves the maximization of the global similarity (*GlobalSIM*), and the satisfaction of the dependencies and restrictions between the features to avoid conflicting and incompatible values. In Fig. 3 is presented a preliminary and simplified algorithm to describe this approach. Finally, the new process can be instantiated from assets corresponding to the selected features.

3.4 Feedback

The learning process in the CBR system is done through the feedback about the performance of the new process model instance, when the project is closed. At this moment, the effectiveness of the reused process is evaluated by the user before the storage in the repository.

In this sense, the assets representation in the repository includes the process feature Success Level to reflect this feedback. This information is useful to the posterior adoption of the process model, and contributes in the search for the continuous improvement of the process.

4 Case Study

In this section, a case study is presented to illustrate the approach for process reuse. In this sense, the description of a new project is detailed assigning values to the wished attributes for process and project. Note that the process for the standard process definition and the instantiation for an already defined process is the same. In the table below the definition of the desired features for the new case are presented. The Scope and Feature columns represent the feature's classification as presented in Table 1. The Weight and Value columns refer to properties of the new project, about the relevance and value for the feature, respectively, from the user viewpoint.

Table 2. Feature definition for the case study.

Scope	Feature	Weight	Value
Project	Life-Cycle Model	Medium	Spiral
	Complexity	Low	Medium
	Size	Medium	Medium
	Team	Medium	5
	Time	Low	6
	SE Knowledge	Low	Medium
	Development Paradigm	High	O-O
Process	Development Model	Low	-
	Maturity Model	Low	-
	Maturity Level	Low	-
	Complexity	Medium	Low
	Process	Medium	Project Management
	Experience on process usage	Low	Low

To illustrate the retrieval process, the RUP for Small Teams (RUP-ST) model [19] and its respective representation are used. It is important to stand out that the repository of process assets must be wide and diversified in order to take care of the most diverse situations. Table 3 presents the values for each feature for a project based on the RUP-ST [19].

The *global similarity* is calculated on the basis of their representation in order to determine and retrieve from the repository the most adherent case to fit the new case through minor efforts.

The *local similarity* for *Feature* is calculated in accordance with the similarity type, as referred to Section 3.2, and is described in the *Comparison* column. The product of this value times the *Weight*, presented in Table 2, determines the *Local Similarity (LS).* Finally, the addition of all local similarities is presented in the column *Global Similarity*, in the current case 345.

Table 3. Global Similarity about RUP-ST.

Scope	Feature	Base-Case	Compa-rison	LS
Project	Life-Cycle Model	Iterative/ Incremental	0,8	40
	Complexity	Medium	1	10
	Size	Medium	1	50
	Team	5	1	50
	Time	5	0,5	5
	SE Knowledge	High	0,5	5
	Development Paradigm	O-O	1	100
Process	Development Model	RUP	0	0
	Maturity Model	-	0	0
	Maturity Level	-	0	0
	Complexity	Medium	0,5	25
	Process	Project Management	1	50
	Experience on process usage	Low	1	10
Global Similarity				**345**

A further analysis about the local similarity results can be used to guide the user during the adaptation process. In this sense, the desired features from the retrieved cases can be composed in a new model in order to optimize (maximize) the global similarity.

To illustrate this approach, the similarities measurements for ProGer [20] and D-CMM [21] models are used in order to select the attributes with higher local similarity value (Table 4). The global similarity result for each base-case indicates the ProGer model as the most similar to the current case, since it presents the greatest measurement value (368,6).

In another side, using the compositional approach, a new model can be obtained on the basis of the maximization algorithm (Fig. 3). The maximized global similarity for the new model, detailed in Table 5, is 383.6. Thus, the model of process created through this approach represents the most adherent (similar) model to the current case, involving lower effort for their adaptation and reuse in the new situation.

Table 4. Global Similarity about ProGer and D-CMM.

Scope	Feature	LS ProGer	LS D-CMM
Project	Life-Cycle Model	40	45
	Complexity	10	10
	Size	50	50
	Team	40	25
	Time	8,6	8,6
	SE Knowledge	10	6,6
	Development Paradigm	100	100
Process	Development Model	0	0
	Maturity Model	0	0
	Maturity Level	0	0
	Complexity	50	25
	Process	50	50
	Experience on process usage	10	5
Global Similarity		**368,6**	**325,2**

The existence of attributes with the same local similarity value is resolved by the selection of the attribute from the first case analyzed; however, is still a need for better research to assess whether this is right. Similarly, the attributes that did not have values for the current case were disregarded, avoiding their influence in the calculation of similarity.

Table 5. Maximizing the Global Similarity.

Feature	Value	Process	SL
Life-Cycle Model	Iterative	D-CMM	45
Complexity	Medium	ProGer	10
Size	Medium	ProGer	50
Team	5	RUP-ST	50
Time	7	ProGer	8,6
SE Knowledge	Medium	ProGer	10
Development Paradigm	O-O	ProGer	100
Complexity	Low	ProGer	50
Process	Project Management	ProGer	50
Experience on process usage	Low	ProGer	10
Global Similarity			**383,6**

The process evolution and improvement is realized along its adaptation, reuse, performance evaluation and reincorporation into the repository. Reuse evaluations along diverse projects can guide the adoption of the organization's standard-process.

5 Final Considerations

The proposed approach promotes the reutilization of process assets as a start point for the elaboration of a standard process to meet the organizational needs. It also can be used to assist in the definition and instantiation of software processes. This approach is based on Case-Based Reasoning. It supplies a mechanism for the representation of cases in the assets repository. The cases are classified according to a set of relevant features to allow an efficient retrieval. An example of similarity measurement was presented. A management tool to support this approach is under development.

In addition, an optimization algorithm for the construction of a new model of process is presented. This model is composed of attributes from different processes, in order to maximize the global similarity, increasing the adherence of the composed process about the new case, and decreasing the adaptation efforts.

This approach foresees the continuous improvement of the process through the permanent feedback to the repository involving the incorporation of learned lessons with the adopted process. The learning capability of CBR systems contribute to the adoption of better and more efficient solutions.

Acknowledgements

This work is being supported in part by FUNCAP, Brazil.

References

1. The International Organization for Standardization and the International Electrotechnical Commission, 1996. Standard for Information Technology—Software Life Cycle Processes. Geneva, Switzerland.
2. The International Organization for Standardization and the International Electrotechnical Commission, 2006. ISO/IEC 15504 Information Technology Process Assessment Part 5.
3. Chrissis, M. Konrad, M., and Shrum, S., 2003. CMMI guidelines for process integration and product improvement. Addison-Wesley.
4. Paulk, M. et al, 1993. Capability Maturity Model for Software. Pittsburg: SEI, Carnegie Mellon University, version 1.1. edition.
5. Softex, 2006. Guia Geral MR-MPS (Versão 1.1). Available in: http://www.softex.br/mpsbr/_guias/MPS.BR_ Guia_Geral_V1.1.pdf
6. PMI Project Management Institute, 2004. A Guide to the Project Management Body of Knowledge: PMBOK Guide. PMI, 3rd edition.
7. Perry, D., 1996. Practical Issues in Process Reuse. In ISPW, International Software Process Workshop. IEEE Computer Society Press. France. Baldonado, M., Chang, C.-C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. Int. J. Digit. Libr. 1 (1997) 108–121.
8. Kolodner, J., 1993. Case-Based Reasoning. Publisher Morgan Kaufmann.
9. Pal, S. and Shiu, S., 2004. Foundation of soft case based reasoning. Wiley series in intelligent systems, 5th ed.
10. Mille, A., 2006. From case-based reasoning to traces-based reasoning. Annual Reviews in Control 30(2):223-232. ELSEVIER. ISSN 1367-5788.
11. Santos V., Cortés M. 2008. Software Process Reuse Using Case-Based Reasoning Accepted for publication in the ICAART´2009. International Conference on Agents and Artificial Intelligence. Portugal.
12. Reis, R. Q., Reis, C.A.L., Nunes, D.J., 2001. Automated Support for Software Process Reuse: Requirements and Early Experiences with the APSEE model. In 7th International Workshop on Groupware. IEEE Computer Society Press. Darmstadt, Germany.
13. Oliveira, K., Gallota, C., Rocha, A. R. et al., 1999. Defining and Building Domain-Oriented Software Development Environments. In ICSSEA'99, 12th International Conference Software & Systems Engineering and their Applications. Paris, France.
14. McManus, J., 1999. How does Software Quality Assurance Fit In. Handbook of Software Quality Assurance, 3 ed. Prentice Hall.
15. Oliveira, S. R. B., Vasconcelos, A. M. L., 2006. A Continuous Improvement Model in ImPProS. In 30th Annual International Computer Software and Applications Conference. Proceedings on COMPSAC Fast Abstract Session. Chicago, EUA.
16. Ricci, F., Arslan, B., Mirzadeh, N., Venturini, A., 2002. Detailed Descriptions of CBR Methodologies. Information Society Technologies. Available in: http://dietorecs.itc.it/PubDeliverables/D4.1-V1.pdf
17. Brasil M., Cortés M. 2008. Definição de Processo de Software através da maximização da similaridade de atributos de casos similares. 13º Simpósio de Informática da PUCRS, Uruguaiana. Revista HÍFEN ISSN 1983-6511.

18. SEI, 2008. CMMI® or Agile: Why Not Embrace Both!. Available in: http://www.sei.cmu.edu/pub/documents/08.reports/08tn003.pdf. Accessed in: 13/11/2008.
19. Pollice, G., Augustine, L., Lowe, C., and Madhur, J., 2004. Software development for small teams - a rup centric approach. Addison-Wesley.
20. Rouiller, A. C. 2001 Gerenciamento de Projetos de Software para Empresas de Pequeno Porte, PhD. Thesis, Universidade Federal de Pernambuco.
21. Orci, T. and Laryd, A. (2000). Dynamic CMM for small organizations. Proceedings of the First Argentine Symposium on Software Engineering (ASSE). Pages 133 to 149. Argentina, 2000.
22. Beck, K. (2004) Extreme Programming Explained: Embrace Change. Pearson.
23. Kruchten P. and Kroll P., 2003. The Rational Unified Process Made Easy. Addison-Wesley.
24. Pressman, R., 2002. Software Engineering, 5th ed. McGraw-Hill.

Appendix

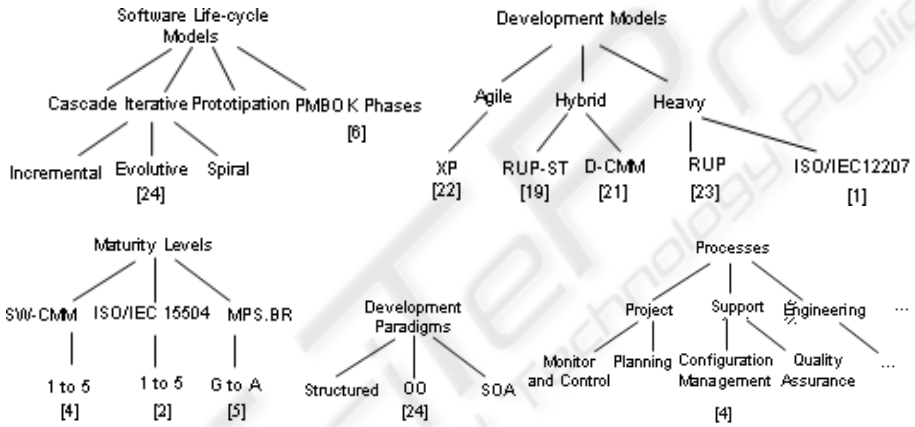

Fig. A1. Taxonomies for QVI features.