On Evidential Feature Salience

George Spanoudakis and Panos Constantopoulos

Department of Computer Science, University of Crete and,
Institute of Computer Science
Foundation of Research and Technology, Hellas

Abstract. This paper describes a method for estimating the salience of features comprising conceptual descriptions of software artifacts. Salience estimates are used in a model analyzing the similarity between such descriptions so as to promote the analogical reuse of the artifacts described by them. Salience is conceived as belief on the dominance of a feature, which is defined on the basis of three general properties that a feature may have in a conceptual model, namely the abstractness, the characteristicity and the causality. This belief is measured according to evidence inherent in conceptual schemas organizing software repositories.

1. Introduction

In this paper, we define a method for evaluating the salience of features comprising conceptual descriptions of software artifacts(i.e code, designs or specifications). The method has been implemented by a system estimating similarities between such descriptions of software artifacts so as to promote their analogical reuse[18].

The analysis of similarity (whose details can be found in [17,18]) results into measures of conceptual resemblance and identifies analogies between software artifacts. Such measures and analogies are used for retrieving artifacts, stored in a Software Information Base[2] and identifying subcomponents of them with prospects for reuse.

The problem of distinguishing between features of different salience arises, while estimating the similarity and detecting analogies between the attributes of two artifact descriptions. Distinctions of salience are critical for analogical reasoning [5,6,7]. Salient features enable the retrieval of source analogs that may lead to consistent and pragmatically usefull conjectures about a target analog, unlike the non salient ones[8,13].

Thus, computational models of analogical reasoning need to estimate the salience of features in the descriptions they operate on. Such estimates are derived either from relevant assertions about features, as in PROTOS[14] and EBL+CBR systems[3], or from assessments of the results of reasoning sessions, provided by external observers(e.g CBL4 algorithms[1]).

As opposed to such estimates, whose derivation is knowledge acquisition intensive, we estimate salience from the conceptual descriptions of artifacts, themselves. The whole approach is based on the concept of *feature dominance*, a compound property derived from three other primitive properties. These are the *characteristicity*, the *abstractness* and the *causality*. All these properties are defined by logical conditions related to the modeling of features over conceptual models, organizing artifact descriptions. Salience is then introduced as belief to the truth value of the dominance of a feature and thus it provides a graded alternative to the logical strictness of it.

The rest of the paper is organized as follows. In the second section, we review the Telos language, which is used for expressing conceptual descriptions of artifacts. In the third section, we discuss and define dominance. In the fourth section, we define the concept of salience and in the fifth section we give an example of estimating salience. Finally, in the sixth section, we summarize the whole approach and review its basic open issues.

2. Telos: A Conceptual Modeling Language for Salience Estimation

Our approach to the estimation of salience is discussed and defined on the basis of the Telos language[12]. Telos offers three modeling abstractions (i.e *multiple and meta classification*, *multiple generalization* and *typed attribution*) which constitute a structurally object oriented data model, expressive enough, to describe software artifacts[2,4,9].

Telos objects are partitioned into *entities* and *attributes* and into *tokens* and *classes*. All objects are described through instantiation relations, have attributes and are uniquelly identified by system identifiers. Class objects are further described through generalization (i.e Isa) relations.

Each Telos class i has an intension INT[i](i.e a set including the identifiers of the attributes it introduces, inherits or refines) and an extension EXT[i](i.e a set including the identifiers of all its instances). Furthermore, for each attribute class i we distinguish its *scope S[i]*(i.e the set of all the classes to which it applies), its *refining classes R[i]* (i.e the set of all the classes introducing or refining it) and its *possible ranges AR[i]*(i.e all the classes/ranges of the attribute, in the classes of its scope). These sets are used for the definition of dominance.

3. Factors of Attribute Dominance

We introduce the concept of dominance to express the importance that an attribute may have in: (i) the construction of conceptual schemas adequately classifying an underlying population of objects with respect to the needs of an application (ii) the existence and the behavior of the objects which are instantiated under the classes to which it applies and, (iii) determining the values of other attributes in its scope. These kinds of importance are realized as the properties of the characteristicity, the abstractness and the causality, respectively.

The Charactericity of Attributes. We define as characteristic (CH_i) any attribute i, which has distinct range classes, for all the classes in its scope, that is $|S[i]| = |AR[i]|$.

Our definition is motivated by the way of forming Isa hierarchies in conceptual modeling. Such hierarchies , consist of classes, which are specialized when particular subsets of their instances have one or more special properties. These properties may be an additional attribute or the restriction of the possible values of an existing attribute. In such cases, the original class may be specialized into a new subclass, which introduces the new attribute or refines the existing one. Refinement is the specialization of the class/range of the attribute by the subclass inheriting it. Introduced and refined attributes are characteristic of the new subclass. Furthermore, to the extent that they are refined in other classes they become characteristic of entire Isa hierarchies and conceptual schemas.

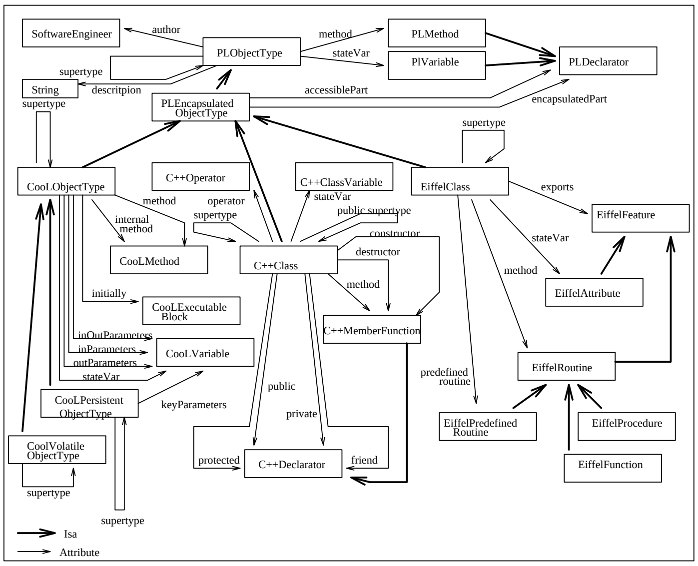

Fig. 1. A Conceptual Model for Object Types of OO Languages

Figure 1 presents characteristic and non characteristic attributes of object types programming languages modeled according to[19]. The attribute *method*, which expresses the methods supported by an object type is characteristic, since it has distinct range classes in all the classes of its scope. Unlike it, the attribute *author* is not characteristic. The way it has been modeled, implies that nothing differentiates the software engineers, who implement Eiffel, CooL and C++ object classes, worth of being abstracted into the conceptual schema.

The Abstractness of Attributes. We also distinguish attributes into abstract and non abstract ones, according to the class which introduces them, in a conceptual schema. In the object oriented data modeling literature, classes are divided into *abstract* and *concrete*, according to whether or not they have any instances of their own [10,11,15]. Abstract classes tend to have attributes, which are essential for the existence and the behavior of the instances of their subclasses.

Consider for example the significance of let's say the attribute *hasSteeringSystem* of a vehicle and the attribute *laggageCarryingCapacity* of a car. The former is essential for driving any sort of vehicle(e.g airplanes, trains, cars), while the latter neither applies to all vehicles(e.g fighting airplanes) nor relates to the functionality of the special kind of vehicles owning it (i.e the cars).

Hence, we define an attribute i to be abstract(i.e ABS_i), if and only if the class which introduces it in a conceptual schema(i.e its *original class*) is abstract. A class j is defined to be abstract (i.e AB_j) if and only if $EXT[j]$=$EXT_s[j]$ where $EXT[j]$ is the extension of j and $EXT_s[j]$ is the *shared extension* of j (i.e the set of its instances, which also belong to one or more of its subclasses).

The Causality of Attributes. We consider as *causal*, attributes whose values, determine the values of other attributes and as such they lead to conjectures about these latter values. For example, the model of a car could dictate the place where it has been produced, if the relevant manufacturer produces its different models, in different countries. This notion of causality has also been realized in the area of explanation based learning, where predefined types of dependency relations are offered in languages for specifying explanations [14]. In order to avoid the cost of acquiring such relations, we restrict our interest in dependencies between attributes that have the same domain and in a special case of them, the *total equivalences*. Both these dependencies have the special property that the conceptual schema may imply their impossibility.

Formally, if two attributes x and y are defined as associations of the following form $x : D_x \rightarrow I_x \quad y : D_y \rightarrow I_y$ where I_x and I_y denote the images(i.e the sets of their actual values) of the attributes x and y, we will say that the attribute y depends on the attribute x if and only if they have identical domains(i.e $D_x \equiv D_y$). A dependency can be defined as a mapping $M_{xy} : I_x \rightarrow I_y$. Total equivalences between x and y, hold in cases where M is a total and onto isomorphism.

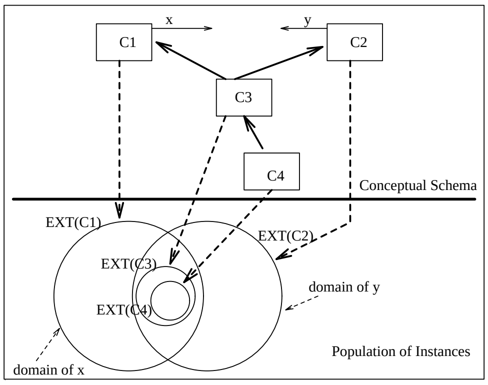

Fig. 2. Non Common Classes of Attributes

The domain equality condition of dependencies is derived from the identity of the scopes of the involved attributes(i.e $S[i] = S[j]$). As it can be observed in figure 2, non common classes in the scopes of two attributes indicate non identical domains(i.e objects in EXT[C1]-EXT[C2] and EXT[C2]-EXT[C1]), unless we assume that all the instances of the class C1 must be also instances of the class C2. However, the presence of two distinct classes in the conceptual schema indicates that their extensions are not identical in general. Otherwise, there would be no need for retaining both classes in the schema (it would be sufficient to have only one class aggregating the attributes of both of them). According to this observation, we will a dependency between the attributes x and y(M_{xy} or $M_{y,x}$) won't be definable whenever $(cnd1)\ :\ S[x] \neq S[y]$.

Futhermore, the definability of total equivalences depends on an additional condition. Given two attributes x and y, a total equivalence mapping between their images cannot be defined if their refining classes do not coincide(i.e $(cnd2)\ :\ R[x] \neq R[y]$).

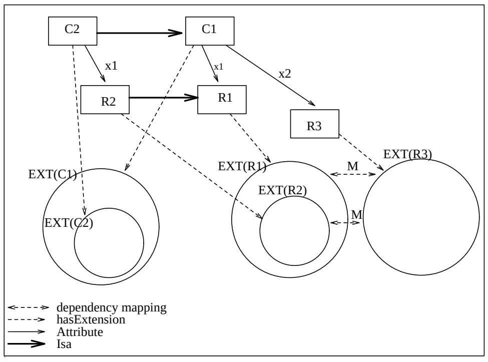

Fig. 3. A Case of Non Common Refinement over the Conceptual Schema

Figure 3, presents a case violating this condition. The the attribute x1 is refined by class C2, unlike the attribute x2. If an equivalence mapping between these two attributes was definable it would be necessary for their images with respect to objects in C1 and to objects in C2, to have equal numbers of elements(i.e $|EXT(R1)| = |EXT(R3)|$ and $|EXT(R2)| = |EXT(R3)|$). Since, the attribute x1 is refined in C2, its image with respect to the objects in C2 will have less elements than its image with respect to objects in C1 (i.e $|EXT(R2)| < |EXT(R1)|$), which contradicts, the previous constraints.

On the basis, of interattribute dependencies, we define an attribute i to be causal (i.e CSL_i) if there can be defined a dependency mapping M_{ij} between its image and the image of any other attribute j, owned by one or more of the classes in its scope.

The Dominance of Attributes. The abstractness, the characteristicity and the causality of attributes determine their dominance. An attribute i, which is characteristic and abstract or causal is defined to be dominant(i.e $DOM_i \equiv (ABS_i$ and CH_i) *or* CSL_i).

The conjunction between the abstractness and the characteristicity trades off between cases satisfying only one of these properties, as extreme ones. In fact, attributes may be introduced in abstract classes only because of their broad applicability. For instance, the attribute *description* of the class *PLObjectType* in figure 1, serves only as a textual annotation of object types with no essential role for the existence or the behavior of object types. Notice also, that attributes introduced in leaf classes of Isa hierarchies, although satisfying our definition of characteristicity, do not have a general classification significance for the entire conceptual schema. These extreme cases can be simultaneously precluded by the logical conjunction between the abstractness and the characteristicity.

In the next section, we present an evidential approximation of dominance, called attribute salience, which relaxes its logical strictness.

4. Salience of Attributes: Belief on their Dominance

We believe that the logical definition of dominance may be proven very strict in cases of incomplete and/or immature conceptual models. It may also be impractical in cases where finner than the two-fold distinctions between dominant and non dominant attributes are required

Thus, we introduce attribute salience as a measure of belief that an attribute is dominant or not. This belief is estimated from partial beliefs about the three properties defining the concept of dominance. Partial beliefs are measured by functions defined over the conceptual schema, and relax the relevant logical definitions.

Our analysis is based on beliefs as they are conceived in the context of the Dempster-Shafer theory[16].

We define salience as a function S such that:

$$S : A \rightarrow [0,...,1] \text{ and } \forall\ i:(i \in A) \rightarrow S(i) \in [Bel(DOM_i),...,P^*(DOM_i)]$$

where A is the set of Telos attribute classes, $Bel(DOM_i)$ is the belief on the dominance of attribute i and, $P^*(DOM_i)$ is the upper probability(or plausibility) of the dominance of attribute i [Sha75].

In the following, we present four distinct belief functions, whose combination estimates these boundaries.

Judging Belief about Charactericity. Successive specializations of classes in a conceptual schema are not always introduced due to refinements of the same attribute. For instance, the class *PLEncapsulatedObjectType* is not introduced due a refinment of the attribute *method*, in figure 1. This observation led us, in a plausible relaxation of the logical definition of characteristicity (i.e $|S[i]| = |AR[i]|$), into a belief measure about this property defined as:

$$m_i^{ch}(P) = \begin{cases} c_i & \text{if } P = CH_i \\ 1-c_i & \text{if } P = \overline{CH_i} \end{cases} \text{ where } c_i = \frac{|AR[i]|}{|S[i]|} .$$

This function results to 1, whenever the defining condition of characteristicity is satisfied.

Judging Belief about Abstractness. Since abstract are only those attributes, which are introduced by abstract classes, the belief to the abstractness of an attribute i is derived from the belief to the abstractness of its original class j. We obtain this latter belief from the function:

$$m_i^{a}(P) = \begin{cases} e_j & \text{if } P = ABS_i \\ 1-e_j & \text{if } P = \overline{ABS_i} \end{cases} \text{ where } e_j = \frac{|EXT_s[j]|}{|EXT[j]|} .$$

This belief function relaxes the definition of abstract classes and gives a belief of 1 whenever this definition is satisfied. Furthermore, it is not as sensitive as that definition to cases where objects are instantiated under abstract classes in a conceptual schema, due to a temporal inability to instantiate them, under some concrete subclass of them.

Judging Belief about Non Causality. Belief about the non causality of an attribute can be measured as an approximate assessment of the conditions *cnd1* and *cnd2* in section 3.

We define the belief to the non existence of a dependency mapping between two attributes i and j, due to their non common classes as:

$$m_{ij}^{ncs}(P) = \begin{cases} d_{ij} & \text{if } P = \overline{M_{ij}} \\ 1-d_{ij} & \text{if } P = \Theta \end{cases} \text{ where } d_{ij} = \frac{(|S[i]-S[j]|+|S[j]-S[i]|)}{(|S[i] \cup S[j]|)}$$

According to $m_{ij}^{ncs}(P)$, the more the non common classes in the scopes of two attributes the less possible the definability of a dependency mapping between them. However, the absence of non common classes does not favour the existence of a dependency mapping, as it is evident by the assignment of the belief $(1-d_{ij})$ to the disjunction M_{ij} *or* $\overline{M_{ij}}$ (i.e Θ).

Also, the belief to the non existence of a total equivalence between two attributes i and j, due to their non common refinements over the schema, is defined as:

$$m_{ij}^{ncr}(P)=\begin{cases} r_{ij} & \text{if } P=\overline{M_{ij}} \\ 1-r_{ij} & \text{if } P=\Theta \end{cases}$$

where $r_{ij} = \dfrac{|(S[i]-R[i])\cap R[j]|+|(S[j]-R[j])\cap R[i]|}{|R[i]\cup R[j]|}$

According to m_{ij}^{ncr}, the more the non common refinements of two attributes over the schema, the less possible the definability of a total equivalence mapping between them. As with the m_{ij}^{ncs} function, the absence of non common refinements does not imply the existence of an equivalence mapping between the involved attributes.

Both the m_{ij}^{ncs} and the $m_{ij}^{ncr}(P)$ result into 0, whenever the negations of the conditions *cnd1* and *cnd2* are satisfied. Thus, they completely disfavour the non existence of dependency or total equivalence mappings between values of attributes.

Total Beliefs about Dominance. The defined partial beliefs can be combined, according to the rule of the orthogonal sum [Sha75] into the following total beliefs:

1. total belief on the dominance of an attribute i(u is the original class of i): $(F1): Bel(DOM_i) = e_u c$

2. total belief on the non causality and upper probability of the causality of an attribute i: $(F2): Bel(\overline{CSL_i}) = \prod_{j=1}^{n} d_{ij}$ and $(F3): Bel(\overline{CSL_i}) = \prod_{j=1}^{n}(1-(1-d_{ij})(1-r_{ij}))$

F2 and F3 apply when all possible dependencies or only total equivalences between attributes are taken into account, respectively.

3. upper probability of the dominance of an attribute i: $(F4): P^*(DOM_i) = (1-(1-c_i e_u)\prod_{j=1}^{n} d_{ij})$ and

$(F5): P^*(DOM_i) = 1-(1-c_i e_u)(\prod_{j=1}^{n}(1-(1-d_{ij})(1-r_{ij})))$

F4 and F5 apply when all possible dependencies or only total equivalences between attributes are taken into account, respectively.

5. An Example

In this section, we give an example of salience estimates, which are computed from the conceptual schema of figure 4. This schema represents, the different types of employees of some research institute. The bold boxes in the figure denote the classes of the taxonomy of employees, and the numbers in their lower right corners are the numbers of the instances of these classes.

Every employee has a social security number(i.e the attribute *securityNumber*) and a legal document determining its employment (i.e the attribute *employmentAgreement*).

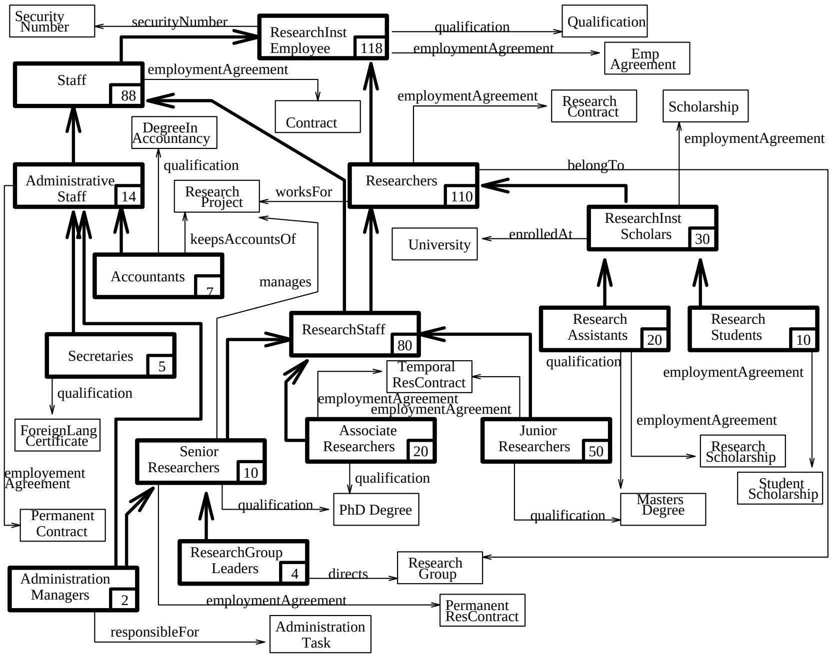

Fig. 4. An Example for Estimating Salience of Atrributes

Also, (s)he needs some typical qualifications(i.e the attribute *qualification*) for occupying a certain position in the institute. Special types of employees may have additional attributes. On the basis of this conceptual schema, the boundaries of the salience of the attributes have been measured and presented in table 1.

The column $Bel(DOM_i)$ gives the lower bound of the total belief on an attribute's dominance. This lower bound may be used as a pessimistic estimate of salience, since it has been estimated without any source of evidence favouring the causality of the attribute in hand(see formula F1). According to it, most salient is the attribute *employmentAgreement*, which expresses the legal determination of the employment for any employee. On the other end of the spectrum, the attribute *keepsAccountsOf* is not essential for the employment, since it applies only to certain types of employees.

Table 1: Lower and Upper Boundaries of Salience							
Attribute	$B(CH_i)$	$B(ABS_i)$	$B_1(\overline{CSL}_i)$	$B_2(\overline{CSL}_i)$	$B(DOM_i)$	$P^*_1(DOM_i)$	$P^*_2(DOM_i)$
employmentAgreement	.53	1	0	.17	.53	1	.92
qualification	.4	1	0	.15	.4	1	.91
enrolledAt	.33	1	.25	.47	.33	.83	.68
manages	.33	.6	.1	.28	.198	.92	.77
worksFor	.1	1	0	0	.1	1	1
belongsTo	.1	1	0	0	.1	1	1
securityNumber	.06	1	0	0	.06	1	1
responsibleFor	1	0	.44	.64	0	.56	.36
directs	1	0	.44	.64	0	.56	.36
keepsAccountsOf	1	0	.81	.77	0	.19	.23

The columns $P_1*(DOM_i)$ and and $P_2*(DOM_i)$ give the upper bounds of the beliefs on dominance. These estimates, obtained from the formulas F4 and F5, could be used as optimistic measures of salience. They only express the degree to which we fail to doubt about the dominance of attributes, according to the interpretation of plausibility by Shafer[Sha75]. According to them, the dominance of the attributes *worksFor* and *belongsTo*, expressing the project and the groups, which a researcher may belong to, could be certain. Both these attributes are not refined separately over the schema and they do not have non common classes. Thus, the existence of a total equivalence mapping between them, cannot be a priori precluded. In fact, it can be intuitively assumed that the participation of a researcher in some research group, determines the project in which (s)he is involved.

6. Concluding Remarks

In this paper, we presented a method for estimating the salience of features comprising descriptions of software artifacts. Salience estimates improve the analysis of similarity between such artifacts[18] which promotes their analogical reuse[17]. Our method only exploits knowledge inherent to conceptual schemas, organizing descriptions of artifacts in repositories. Thus, it is expected to be more practical than other knowledge acquisition intensive solutions to the salience estimation problem, which are incorporated in computational models of analogical reasoning. The described method could be applied to various organizational schemas for repositories, provided that they are expressible through classification, generalization and attribution relations, which are assumed in our framework.

Further research will concentrate on the evaluation of the method in software repositories organized according to diverse conceptual schemas and will investigate into the completeness of the properties used for defining dominance.

7. References

[1] Aha D., Case-Based Learning Algorithms, Proceedings of the DARPA Case-Based Reasoning Workshop, 1991

[2] Constantopoulos P., Doerr M., Vassiliou Y., Repositories for Software Reuse: The Software Information Base, In Proceedings of the IFIP Conference in the Software Development Process, Como,Italy, 1993

[3] Cain T., et al., Using Domain Knowledge to Influence Similarity Judgements, Proceedings of the DARPA Case-Based Reasoning Workshop, 1991

[4] Devanbu P. et al., LaSSIE: A Knowledge Based Software Information System, CACM 34(5), 1991

[5] Falkenhainer B., et al., The Structure Mapping Engine: Algorithm and Examples, Artificial Intelligence, 41, 1989

[6] Gentner D., Finding the Needle: Accessing and Reasoning From Prior Cases, Proceedings of the DARPRA Workshop on Case-Based Reasoning, 1988

[7] Hall R., Computational Approaches to Analogical Reasoning: A Comparative Analysis, Artificial Intelligence 39, 1989

[8] Holyoak K., Koh K., Surface and Structural Similarity in Analogical Transfer, Memory and Cognition 15(4), 1987

[9] Johnson L., et al., Representation and Presentation of Requirements Knowledge, IEEE Transactions on Software Engineering 18(10), 1992

[10] Johnson R., Designing Reusable Classes, Journal of Object Oriented Programming, June/July 1988

[11] Meyer B., Object Oriented Software Construction, C.A.R Hoare, Series Editor, 1989

[12] Mylopoulos J., et. al., Telos: Representing Knowledge About Information Systems, ACM Transactions on Information Systems, 8(4), 1990

[13] Novick L., Analogical Transfer: Processes and Individual Differences, Analogical Reasoning, Kluwer Academic Publishers, 1988

[14] Porter W.B., et al., Concept Learning and Heuristic Classification in Weak-Theory Domains, Artificial Intelligence, 45, 1990

[15] Sciore E., Object Specialization, ACM Transactions on Office Information Systems, 7(2), 1989

[16] Shafer G., A Mathematical Theory of Evidence, Princeton University Press, 1975

[17] Spanoudakis G., Constantopoulos P., Similarity for Analogical Software Reuse: A Conceptual Modeling Approach, Proceedings of the 5th Conference on Advanced Information Systems Engineering(CAiSE ’93), 1993

[18] Spanoudakis G., Constantopoulos P.,Measuring Similarity Between Software Artifacts, Proceedings of the 6th International Conference on Software Engineering and Knowledge Engineering(SEKE ’94, 1993

[19] Weber M., Implementation Models for the SIB, ITHACA.SNI.92.E2.#2, September 1992