Journal of Universal Computer Science, vol. 16, no. 20 (2010), 3138-3162
submitted: 7/1/10, accepted: 13/9/10, appeared: 1/11/10 © J.UCS

Extending the Methods for Computing the Importance of Entity Types in Large Conceptual Schemas

Antonio Villegas
(Universitat Politècnica de Catalunya – Barcelona, Spain
antonio.villegas@upc.edu)

Antoni Olivé
(Universitat Politècnica de Catalunya – Barcelona, Spain
antoni.olive@upc.edu)

Abstract: Visualizing and understanding large conceptual schemas requires the use of specific methods. These methods generate clustered, summarized, or focused schemas that are easier to visualize and understand. All of these methods require computing the importance of each entity type in the schema. In principle, the totality of knowledge defined in the schema could be relevant for the computation of that importance but, up to now, only a small part of that knowledge has been taken into account. In this paper, we extend seven existing methods for computing the importance of entity types by taking into account more relevant knowledge defined in the structural and behavioural parts of the schema. We experimentally evaluate the original and extended versions of these methods with three large real-world schemas. We present the two main conclusions we have drawn from the experiments.

Key Words: Conceptual Modeling, Large Schemas, Visualization
Category: H.0, H.1, H.3.3, H.5

1 Introduction

Real information systems often have extremely complex conceptual schemas. Visualizing and understanding these schemas requires the use of specific methods, which are not needed in small schemas [Olivé and Cabot (2007)]. These methods generate indexed, clustered, summarized, or focused schemas that are easier to visualize and understand [Lindland et al. (1994)].

Many of the above methods require computing the *importance* (also called *relevance* or *score*) of each type in the schema. Each method has its own definition of entity-type importance. In each method, the computed importance induces an ordering of the entity types, which plays a key role in the steps and result (output) of the method. For example, Castano, de Antonellis, Fugini and Pernici [Castano et al. (1998)] propose a three-step indexing method, in which the first step computes the importance of each entity type, based on the number and kind of relationships it has in the schema. Moody and Flitman [Moody and Flitman (1999)] propose a clustering method in which the most important entity types are hypothesized to be those that have higher connectivity, defined as the number of relationships in which they participate. Tzitzikas and Hainaut propose methods for scoring each entity type in a schema [Tzitzikas and Hainaut (2005), Tzitzikas et al. (2007)], aiming at facilitating its understanability. As a last example, we mention the work of Yu and Jagadish [Yu and Jagadish (2006)], who propose a metric of the importance of each entity type, which is used in order to automatically produce a good schema summary.

As far as we know, existing metrics for entity type importance are mainly based on the amount of knowledge defined in a schema, but only take into account the number of attributes, associations, and generalization/specialization relationships. Surprisingly, none of the methods we found in the literature take into account additional knowledge about entity types defined in a schema that, intuitively, could have an effect on importance. A complete schema [Olivé (2007)] also includes cardinalities, taxonomic constraints, general constraints, derivation rules, and the specification of events, all of which contribute to knowledge about entity types.

In this paper we focus on objective metrics, which are independent from subjective evaluations of users and modelers. Intuitively, it seems that an objective metric of the importance of an entity type in a given schema should be related to the amount of knowledge that the schema defines about it. The more (less) knowledge a schema defines about an entity type, the more (less) important that entity type should be in the schema. Adding more knowledge about an entity type should increase (or at least not decrease) the relative importance of that entity type with respect to others. This assumption is also found in other fields like text information retrieval, document sorting, and web search engines where the frequency of word occurrences and the number of links between documents are indicators of their importance [Salton (1989)].

The main objective here is to analyze the influence of that additional knowledge on a representative set of existing methods for measuring the importance of entity types. To this end, we have selected seven methods from the literature and have developed extended versions of all of them[1]. We do not essentially change the way they define importance: we just add more elements in their computation. We have experimentally evaluated both versions of each method using the conceptual schema of osCommerce [Tort and Olivé (2007)], the UML metaschema [OMG (2009)] and the conceptual schema of EU-Rent [Business Rules Group (2000), Frias et al. (2003)].

OsCommerce is a popular industrial e-commerce system whose conceptual schema consists of 346 entity types (of which 261 are event types). The official UML 2.0 metaschema [Bauerdick et al. (2004), Bauerdick et al. (2004b)] we have used includes 293 entity types. The EU-Rent (a car rental company) case study

[1] This paper is an extended version of [Villegas and Olivé (2009)].

contains 185 entity types (of which 120 are event types).

In our implementation, the original and extended methods give exactly the same results from the same input, but the extended versions can process additional knowledge defined in the schema and then, of course, they give different results. We analyze the differences, and make conclusions on the effect of the additional knowledge on the metrics.

The rest of the paper is organized as follows. [Section 2] introduces the concepts and notations. [Section 3] briefly describes the seleted methods and explains our extensions. [Section 4] describes the experimentation with the methods, the results obtained and the conclusions we have drawn. Finally, [Section 5] summarizes the paper and points out future work.

2 Basic Concepts and Notations

In this section we review the main concepts and notations we have used to define the knowledge of conceptual schemas and the different measures that extract information from them, to be used in the importance computation methods.

2.1 Basic Measures

The conceptual schema of an information system is the general knowledge that the information system needs to know in order to perform its functions. A conceptual schema comprises general knowledge about the domain and knowledge about the functions to be performed [Olivé (2007)].

A conceptual schema consists of a structural (sub)schema and a behavioral (sub)schema. The structural schema consists of a taxonomy of entity types (a set of entity types with their generalization/specialization relationships and the taxonomic constraints), a set of relationship types (either attributes or associations), the cardinality constraints of the relationship types, and a set of other static constraints. In this paper, we deal with schemas written in the UML/OCL [OMG (2009), OMG (2006)].

Furthermore, entity and relationship types may be base or derived. If they are derived, there is a formal derivation rule in OCL that defines their population in terms of the population of other types.

The behavioural schema of a conceptual schema consists of a set of event types. We adopt the view that events can be modeled as a special kind of entity type. Event types have characteristics, constraints and effects. The characteristics of an event are its attributes and the associations in which it participates. The constraints are the conditions that events must satisfy to occur.

Each event type has an operation called *effect()* that gives the effect of an event occurence. The effect is declaratively defined by the postcondition of the operation, which is specified in OCL [Olivé and Raventós (2006)].

[Tab. 1] summarizes the notation of the basic metrics to extract knowledge from the conceptual schema (inspired by [Tzitzikas et al. (2007), Baroni (2002)]) used in the rest of the paper.

We denote by $\mathcal{E}$ the set of entity types defined in the schema. For a given $e \in \mathcal{E}$ we denote by $par(e)$ and $chi(e)$ the set of direct ascendants and descendants of e, respectively, and by $gen(e)$ the union of both sets. For example [see Fig. 1], if each instance of the entity type *Part* is an instance of *Item* (there is a specialization relationship from *Item* to *Part*), we have that $Item \in par(Part)$ and $Part \in chi(Item)$, and in this example $par(Part) = \{Item\}$ and $chi(Item) = \{Part\}$.

The set of attributes defined in the schema is denoted by $\mathcal{A}$. If $a \in \mathcal{A}$ then $entity(a)$ denotes the entity type where a is defined. The set of attributes of an entity type e is denoted by $attr(e)$. If *Item* has the attribute *name*, then $entity(name)$=*Item* and $attr(Item)$=$\{name\}$.

Notation	Definition
$par(e)$	$= \{e' \in \mathcal{E} \mid e\ IsA\ e'\}$
$chi(e)$	$= \{e' \in \mathcal{E} \mid e'\ IsA\ e\}$
$gen(e)$	$= par(e) \cup chi(e)$
$attr(e)$	$= \{a \in \mathcal{A} \mid entity(a) = e\}$
$members(r)$	$= \{e \in \mathcal{E} \mid e \text{ is a participant of } r\}$
$assoc(e)$	$= \{r \in \mathcal{R} \mid e \in members(r)\}$
$conn(e)$	$= \uplus_{r \in assoc(e)}\{members(r) \backslash \{e\}\}^2$
$par_{inh}(e)$	$= par(e) \cup \{par_{inh}(e') \mid e' \in par(e)\}$
$chi_{inh}(e)$	$= chi(e) \cup \{chi_{inh}(e') \mid e' \in chi(e)\}$
$attr_{inh}(e)$	$= attr(e) \cup \{attr_{inh}(e') \mid e' \in par(e)\}$
$assoc_{inh}(e)$	$= assoc(e) \uplus \{assoc(e') \mid e' \in par_{inh}(e)\}$
$conn_{inh}(e)$	$= conn(e) \uplus \{conn(e') \mid e' \in par_{inh}(e)\}$

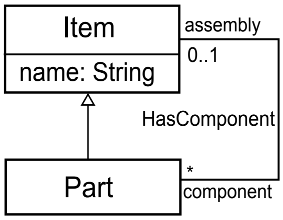
Table 1: Definition of basic metrics.

The set of relationship types defined in the schema is denoted by $\mathcal{R}$. If $r \in \mathcal{R}$ then $members(r)$ denotes the set of entity types that participate in the relationship r, and $assoc(e)$ the set of associations in which e participates. Note that an entity type e may participate more than once in the same association, and therefore $members(r)$ and $assoc(e)$ are multisets (may contain duplicate elements). Moreover, $conn(e)$ denotes the multiset of entity types connected to e through associations [see Fig. 1].

The last row section in [Tab. 1] defines the notation we use to take into account the inherited properties from the ancestors of entity types. Conceptually,

[2] Note that "\" denotes the difference operation of multisets as in $\{a, a, b\}\backslash\{a\} = \{a, b\}$ and "$\uplus$" denotes the multiset (or bag) union that produces a multiset as in $\{a, b\} \uplus \{a\} = \{a, a, b\}$

each entity type e inherits the attributes and participations in relationships of all its parents (those in $par_{inh}(e)$). We denote by $attr_{inh}(e)$ and $assoc_{inh}(e)$ the attributes and participations in associations inherited by the entity type e from all its (direct and indirect) parents through generalization/specialization relationships. In addition to it, $conn_{inh}(e)$ is the multiset of the entity types directly connected to e but also including those entity types connected to its parents ($par_{inh}(e)$), through associations. A special case, $chi_{inh}(e)$ is the set of all the descendants of e.



par(Item) = Ø
chi(Item) = {Part}
gen(Item) = {Part}
attr(Item) = {name}
assoc(Item) = {HasComponent}
conn(Item) = {Part}

par_{inh}(Item) = Ø
chi_{inh}(Item) = {Part}
$attr_{inh}$(Item) = {name}
$assoc_{inh}$(Item) = {HasComponent}
$conn_{inh}$(Item) = {Part}

par(Part) = {Item}
chi(Part) = Ø
gen(Part) = {Item}
attr(Part) = Ø
assoc(Part) = {HasComponent}
conn(Part) = {Item}

par_{inh}(Part) = {Item}
chi_{inh}(Part) = Ø
$attr_{inh}$(Part) = {name}
$assoc_{inh}$(Part) = {HasComponent, HasComponent}
$conn_{inh}$(Part) = {Item, Part}

members(HasComponent) = {Item, Part}

Figure 1: Example of basic metrics.

2.2 Extended Knowledge

This section presents the additional knowledge we have included in the methods to compute the importance of entity types.

2.2.1 Schema Rules

Most existing methods for computing the importance of entity types only take into account the number of attributes, associations, and generalization/specialization relationships. However, there are other schema elements that involve entity types and that could be taken into account when computing importance. These elements are integrity constraints, derivation rules, event constraints, and event postconditions. We call these four kinds of schema elements *schema rules*. In this paper, we assume that the schema rules are written in OCL.

Each schema rule is defined in the context of an entity type, and its expression involves entity types, attributes, and associations. For example, the schema rule (constraint in this case):

```
context Employee inv minSalaryRule:
   self.salary >= self.company.industry.minSalary
```

is defined in the context of the entity type Employee (see [Fig. 2]) and its expression involves entity types *Employee*, *Company*, *Industry*, associations *WorksFor*, *OperatesIn*, and attributes *Employee::salary* and *Industry::minSalary*.

We approach the problem of taking into account the schema rules in the computation of the importance of entity types by considering that a schema rule defines a set of binary links (that we call rule links) between entity types, which can then be considered in a similar way to that of ordinary binary associations, attributes, or generalization/specialization relationships.

There are two kinds of rule links: *Context-Participant* (CP) and *Participant-Participant* (PP) links. There is a CP link in a schema rule between entity types e and e' if e is the context of that rule and e' participates in the expression of that rule. There is a PP link in a schema rule between entity types e and e' if in the expression of that rule there is a navigation through a binary association (or attribute) whose participants are e and e'.

The example in [Fig. 2] illustrates our approach with a simple schema containing three entity types and two relationships. The top of the figure describes a schema rule that constraints the salary of employees. Concretely, the salary of an employee must be greater than or equal to the minimum salary of the industry of his or her company. Furthermore, [Fig. 2] shows a breakdown of the different sections of the OCL expressions to show the referenced entity types.

It is possible to distinguish between the entity types that participate in the body of this schema rule (indicated with dashed lines) and the context entity type (indicated with a dotted line). Also, two arrows indicate the structural navigations through relationship types in the schema rule (from `self`, i.e. *Employee*, to *Company*, and from *Company* to *Industry*).

It is important to note that although an entity type may participate more than once in the same body of a schema rule, to avoid repetitions only one CP link is created between the context entity type and the participant entity type. The bottom part of [Fig. 2] shows the three CP links in the example (dashed lines (a), (b) and (c)).

We also create PP links between the participants that appear as source and target in a navigation expression of the OCL. In the example of [Fig. 2], the arrows of the top indicate two structural navigations. By structural navigations we mean navigations through relationship types. In this case we have the OCL expression `self.company`, which indicates a navigation through the relationship between *Employee* and *Company* (`self` references the context of the schema rule, in this case, *Employee*). Also we have `company.industry` that shows the navigation through the relationship between *Company* and *Industry*. In the bottom part of [Fig. 2] the PP links are shown (dotted lines (d) and (e)).

Finally, once we have extracted the CP and PP links from the schema rule, the next step consists of removing repeated links. Therefore, the final set of rule

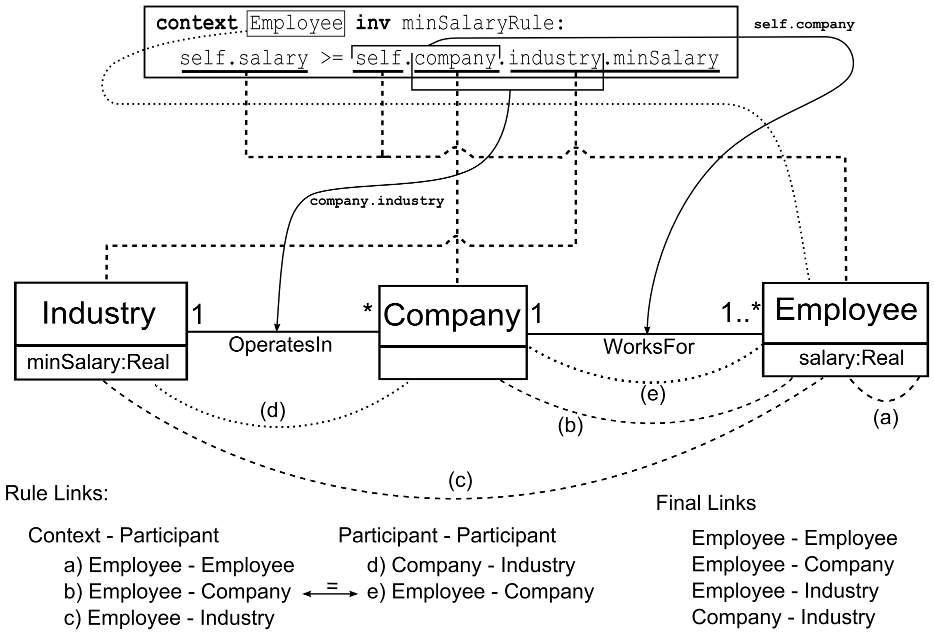


Figure 2: Example of rule links extracted from the OCL.

links from a schema rule is the union of both sets. This way, the repeated link *Employee*-*Company* in the example will appear only once. [Fig. 2] shows the four links extracted from the `minSalaryRule` rule.

2.2.2 Cardinality Constraints

It is well known that cardinality constraints are some of the most important constraints in a schema, and that they are usually defined by means of a special graphical notation (multiplicities in the UML). Surprisingly, as far as we know, cardinality constraints have not been taken into account by existing methods for computing the importance of entity types.

In this paper, we consider cardinality constraints to be general knowledge and thus the presence of such knowledge in a schema should increase the importance of the entity types involved in these constraints. In our approach, we do not give special treatment to cardinality constraints. Instead, we transform them into equivalent OCL expressions, and then we treat them like ordinary schema rules. For example, assume the association between *Industry* and *Company* of [Fig. 2], with the cardinality constraint that a company belongs to exactly one industry.

Its equivalent is the OCL schema rule:

```
context Company inv: self.industry->size()=1
```

This rule generates CP links between *Company* and *Industry* and between *Company* and itself. It also generates a PP link between *Company* and *Industry* (because of the navigation through the association *OperatesIn*). Therefore, the final links are *Company-Company* and *Company-Industry*.

On the other hand, the multiplicity "*" in the company role of the association *OperatesIn* ([Fig. 2]) means that the number of companies that operate in a given industry is unconstrained, and therefore that multiplicity is ignored and not transformed into any schema rule.

2.2.3 Taxonomic Constraints

As far as we know, existing methods for computing the importance of entity types do not take into account the well-known taxonomic constraints (*disjoint* and *complete* in UML).

In our approach we transform these constraints into their OCL equivalents, and we treat them like ordinary schema rules. For example, if there is a disjointness constraint between *Man* and *Woman*, the OCL expressions are:

```
context Man inv: not self.oclIsTypeOf(Woman)
context Woman inv: not self.oclIsTypeOf(Man)
```

The first rule generates the CP links Man-Man and Man-Woman. Similarly, the second rule generates the CP links Woman-Woman and Woman-Man.

2.2.4 Association Classes

The association class is a well-known conceptual modeling construct that is also part of UML. As we have done with cardinality and taxonomic constraints, we do not provide a specific treatment for association classes but, instead, transform them into other constructs. We follow the transformation approach described in [Snoeck and Dedene (1998)]. It consists of removing the association A of the association class and adding new binary associations between the entity type of the association class and the participant entity types of the association A. The cardinality constraints of the new binary relationships must maintain the same semantics as before the transformation.

In the case of a binary association class (left side of [Fig. 3]), the cardinality constraints after the transformation (right side of [Fig. 3]) are interchanged between the participants of the relationship and the association class. The cardinality constraints of a participant (e.g. $a_{min}...a_{max}$ of A in left side of [Fig. 3]) go to the new binary relationship between the other participant (B) and the new entity (C) representing the previous association class, placed on the side of the new entity type. On the side of the previous participants (A and B) the cardinality equals 1.

Furthermore, to maintain the semantics we also need a new uniqueness constraint. An example of a uniqueness constraint is shown in [Fig. 3] for the transformation of the association class *C*. Note that the constraint is specified in OCL.

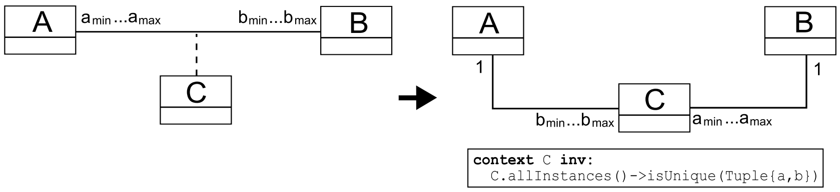


Figure 3: Transformation of a binary relationship with association class.

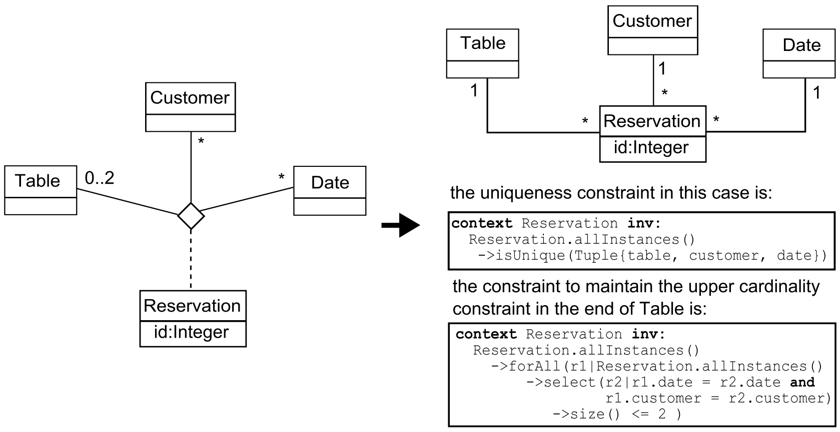


Figure 4: Transformation of a ternary relationship with association class.

For the case of an n-ary association class the transformation is similar (see [Fig. 4]). However, the multiplicities in the new binary relationships between the participants and the entity type representing the association class are always "1" on the participant side and "*" on the entity-type side.

In this case a uniqueness constraint is also needed and follows the same idea as with binary association classes. Furthermore, the cardinality constraints of the initial association must be expressed as an OCL constraint to maintain the semantics. For example, the multiplicity “0..2” in the *Table* participant of [Fig. 4] is transformed into the OCL constraint shown at the bottom of this figure.

2.3 Extended Measures

Next, we will introduce new metrics that will take into account the extracted rule links from each one of the schema rules of the conceptual schema [see Tab. 2].

We denote by $\mathcal{SR}$ (Schema Rules) the set of constraints, derivation rules, and pre- and postconditions. Each rule $sr \in \mathcal{SR}$ is defined in the context of an entity type, denoted by $context(sr)$. In OCL, each rule sr consists of a set of OCL expressions (see [OMG (2006)]) which we denote by $expr(sr)$. An expression exp may refer to several entity types which are denoted by $members(exp)$. The set of entity types that are referenced in one or more expressions of a rule sr is denoted by $ref(sr)$.

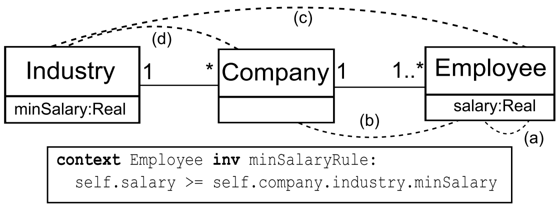
A special kind of OCL expression is the navigation expression that defines a navigation in the schema from one entity type to another through an association (see *NavigationCallExp* of OCL in [OMG (2006)]). We use $expr_{nav}(sr)$ to indicate the navigation expressions inside a rule $sr \in \mathcal{SR}$. Such expressions only contain two entity types as its participants, i.e. the *source* entity type and the *target* one [see the example in Fig. 5].

Notation	Definition
$members(exp)$	$= \{e \in \mathcal{E} \mid e \text{ is a participant of } exp\}$
$expr(sr)$	$= \{expr \mid expr \text{ is contained in } sr\}$
$ref(sr)$	$= \cup_{exp \in expr(sr)}\{members(exp)\}$
$expr_{nav}(sr)$	$= \{expr \in expr(sr) \mid expr \text{ is a navigation expression}\}$
$nav_{expr}(sr)$	$= \cup_{exp \in expr_{nav}(sr)}\{\{e, e'\} \subset \mathcal{E} \mid \{e, e'\} = members(exp)\}$
$nav_{context}(sr)$	$= \{\{e, e'\} \subset \mathcal{E} \mid e = context(sr) \wedge e' \in ref(sr)\}$
$nav(sr)$	$= nav_{context}(sr) \cup\ nav_{expr}(sr)$
$rconn(e)$	$= \uplus_{sr \in \mathcal{SR}}\{e' \in \mathcal{E} \mid \{e, e'\} \subset nav(sr)\}^3$
$rconn_{inh}(e)$	$= rconn(e) \uplus \{rconn_{inh}(e') \mid e' \in par(e)\}$

Table 2: Definition of extended metrics.

[3] We require $rconn(e) = \emptyset$ (empty set) if $conn_{inh}(e) = \emptyset$.

We denote by $nav_{expr}(sr)$ the set of pairs that participate in the navigation expressions of sr. We also denote by $nav_{context}(sr)$ the sets of pairs of entity types composed by the context of the rule sr and every one of the participant entity types of the rule ($e \in ref(sr)$). Finally, we define $nav(sr)$ as the union of $nav_{context}(sr)$ with $nav_{expr}(sr)$, and $rconn(e)$ as the multiset of entity types that compose a pair with e in $nav(sr)$. Note that since we use $\uplus$, $rconn(e)$ may contain duplicates because it takes into account that each rule sr and an entity type e can be related to another one e' in two or more different rules. Intuitively, $rconn(e)$ is the multiset of entity types to which an entity type e is connected through schema rules.



$$
\begin{aligned}
context(\texttt{minSalaryRule}) &= \text{Employee} \\
expr_{nav}(\texttt{minSalaryRule}) &= \{\texttt{self.company}, \texttt{company.industry}\} \\
ref(\texttt{minSalaryRule}) &= \{\text{Employee, Company, Industry}\} \\
nav_{context}(\texttt{minSalaryRule}) &= \{\{\text{Employee, Employee}\}, \{\text{Employee, Company}\}, \\
&\quad\; \{\text{Employee, Industry}\}\} \\
nav_{expr}(\texttt{minSalaryRule}) &= \{\{\text{Employee, Company}\}, \{\text{Company, Industry}\}\} \\
nav(\texttt{minSalaryRule}) &= \{\{\text{Employee, Employee}\}, \{\text{Employee, Company}\}, \\
&\quad\; \{\text{Employee, Industry}\}, \{\text{Company, Industry}\}\} \\
&\textit{if we only take into account the invariant } \texttt{minSalaryRule}\textit{:} \\
rconn(\text{Industry}) &= \{\text{Company, Employee}\} \\
rconn(\text{Company}) &= \{\text{Employee, Industry}\} \\
rconn(\text{Employee}) &= \{\text{Employee, Company, Industry}\}
\end{aligned}
$$

Figure 5: Example of navigations of the schema rule `minSalaryRule`.

Dashed lines (a), (b) and (c) in [Fig. 5] represent the Context-Participant links in $nav_{context}$(`minSalaryRule`) while (b) and (d) are the Participant-Participant connections through navigation expressions (see nav_{expr}(`minSalaryRule`)).

3 Methods and their Extensions

In this section we briefly review the definition of seven existing methods for computing the importance of entity types in a schema. Each method is followed by a brief description and formal definition of our extension to it. An extended review of these methods is presented in [Villegas (2009)].

The original version of the methods only takes into account the indicated elements of the structural schema while in the extended version we also take into account the rules and the complete behavioural schema.

[Fig. 6] shows an example of conceptual schema that will be used to illustrate the methods.

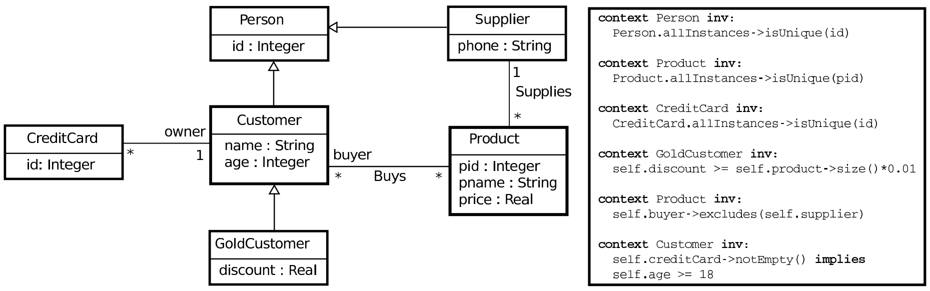


Figure 6: Example of conceptual schema with some OCL invariants.

3.1 The Connectivity Counter

The first method we present was introduced in [Moody and Flitman (1999)]. They suggest that central entities should be chosen as the entities of highest business importance —the *core* business concepts in the model.

Although business importance is quite a subjective concept, and therefore it would require human judgment, Moody and Flitman provide a useful heuristic for identifying central entities in an objective manner. It consists in identifying entities with the most relationships. The authors state that the most highly connected entities are also the most important entities from a business viewpoint.

Following these indications we define the Connectivity Counter (CC) method as the method that computes the importance (I_{CC}) of each entity type as the number of relationships it participates in. Formally:

$$I_{CC}(e) = |assoc(e)|^4$$

Our extension to this method (CC+) follows the same idea as the original version but also including the number of participations of each entity type in the rule links extracted from the schema rules specification. On the other hand, we now

[4] $|A|$ indicates the total number of elements (cardinality) of a set A. When A is a multiset then $|A|$ takes into account repeated elements. For example, if $A = \{e_1, e_1, e_2\}$ then, $|A| = 3$.

take into account (in $|assoc(e)|$) the associations of each entity type e with the event types of the behavioural schema (in case of such events were defined). Formally:

$$I^+_{CC}(e) = |assoc(e)| + |rconn(e)|$$

[Tab. 3] presents the results of these methods for the example in [Fig 6]. Note that CC+ gives more importance to *Product* than to *Customer*, as opposed to CC, because it has more connections in rules.

e	$\|assoc(e)\|$	$\|rconn(e)\|$	$\boldsymbol{I_{CC}(e)}$	$\boldsymbol{I^+_{CC}(e)}$
CreditCard	1	6	**1**	**7**
Customer	2	10	**2**	**12**
GoldCustomer	0	4	**0**	**4**
Person	0	0	**0**	**0**
Product	2	14	**2**	**16**
Supplier	1	4	**1**	**5**

Table 3: Results for CC and CC+ applied to example of [Fig 6].

3.2 The Simple Method

The Simple Method was introduced in [Tzitzikas et al. (2007)] and takes into account only the number of directly connected elements of each entity type to compute its importantce. Formally, the importance $I_{SM}(e)$ of an entity type e is defined as:

$$I_{SM}(e) = |par(e)| + |chi(e)| + |attr(e)| + |assoc(e)|$$

As in the extended version of the Connectivity Counter, the Simple Method has been extended including the number of participations of each entity type in rule links extracted from the schema rules. We also take into account (in $|assoc(e)|$) the associations of each entity type e with the event types of the behavioural schema (in case of such events were defined). Formally:

$$I^+_{SM}(e) = |par(e)| + |chi(e)| + |attr(e)| + |assoc(e)| + |rconn(e)|$$

For example, in the schema shown in [Fig. 5] we would have $I_{SM}(Company)$ = 2 and for the extended version $I^+_{SM}(Company)$=8, because $|par(Company)|$ = $|chi(Company)|$ = $|attr(Company)|$ = 0, $|assoc\ (Company)|$=2, and the new component $|rconn(Company)| = 7$, of which two come for the invariant (`minSalaryRule`) and the other five from the OCL equivalent to the cardinality constraints in its relationships with *Industry* and *Employee*.

e	$\|par(e)\|$	$\|chi(e)\|$	$\|attr(e)\|$	$\|assoc(e)\|$	$\|rconn(e)\|$	$\boldsymbol{I_{SM}(e)}$	$\boldsymbol{I^{+}_{SM}(e)}$
CreditCard	0	0	1	1	6	**2**	**8**
Customer	1	1	2	2	10	**6**	**16**
GoldCustomer	1	0	1	0	4	**2**	**6**
Person	0	2	1	0	0	**3**	**3**
Product	0	0	3	2	14	**5**	**19**
Supplier	1	0	1	1	4	**3**	**7**

Table 4: Results for SM and SM+ applied to example of [Fig 6].

The values obtained after the application of the simple method (and its extended version) to the previous schema shown at [Fig 6] are indicated in [Table 4]. There, it is shown that the most important entity types are *Customer* and *Product*. Also note that *CreditCard* increases its importance in the extended version with respect to the original one due to its presence in rules.

3.3 The Weighted Simple Method

The Weighted Simple Methos is a variation to the Simple Method that assigns a relevance coefficient to each kind of component of knowledge in the equation, such that the higher the relevance, the greater the importance of such component [Castano et al. (1998)]. The definition of importance here is:

$$I_{WSM}(e) = q_{inh}(|par(e)| + |chi(e)|) + q_{attr}|attr(e)| + q_{assoc}|assoc(e)|$$

where q_{attr} is the coefficient for attributes, q_{inh} is the coefficient for generalization/specialization relationships, and q_{assoc} is the coefficient for associations. Each of them with values in the interval [0,1].

Our extension to this method (WSM+) consists in adding the rule links component to the importance computation. In the same way as the other components, we selected a coefficient (q_{rule}) to specify the relevance of rule links of the schema rules. The definition is now:

$$\begin{aligned} I^{+}_{WSM}(e) = {} & q_{inh}(|par(e)| + |chi(e)|) \\ & + q_{attr}|attr(e)| + q_{assoc}|assoc(e)| + q_{rule}|rconn(e)| \end{aligned}$$

The results of the application of the weighted simple method and its extended version (with the same coefficients as in [Castano et al. (1998)] and $q_{rule} = q_{assoc}$) to the example in [Fig 6] are shown in [Tab. 5].

3.4 The Transitive Inheritance Method

The Transitive Inheritance Method [Tzitzikas et al. (2007)] is a variation of the Simple Method taking into account both directly defined features and inherited

e	$q_{inh}(\|par(e)\|$ $+\|chi(e)\|)$	q_{attr} $\|attr(e)\|$	q_{assoc} $\|assoc(e)\|$	q_{rule} $\|rconn(e)\|$	$\boldsymbol{I_{WSM}(e)}$	$\boldsymbol{I^{+}_{WSM}(e)}$
CreditCard	0.6x0	1x1	0.4x1	0.4x6	**1.4**	**3.8**
Customer	0.6x2	1x2	0.4x2	0.4x10	**4**	**8**
GoldCustomer	0.6x1	1x1	0.4x0	0.4x4	**1.6**	**3.2**
Person	0.6x2	1x1	0.4x0	0.4x0	**2.2**	**2.2**
Product	0.6x0	1x3	0.4x2	0.4x14	**3.8**	**9.4**
Supplier	0.6x1	1x1	0.4x1	0.4x4	**2**	**3.6**

Table 5: Results for WSM and WSM+ applied to example of [Fig 6].

ones . For each entity type the method computes the number of ascendants and descendants and all specified attributes and accessible associations from it or any of its ascendants. Formally:

$$I_{TIM}(e) = |par_{inh}(e)| + |chi_{inh}(e)| + |attr_{inh}(e)| + |assoc_{inh}(e)|$$

In the same way as before, we extend it with the rule links component. This time the computation of such component also takes into account the *rconn* of the ancestors:

$$I^{+}_{TIM}(e) = |par_{inh}(e)| + |chi_{inh}(e)| + |attr_{inh}(e)| + |assoc_{inh}(e)| + |rconn_{inh}(e)|$$

The results of the application of the TIM+ method to the example in [Fig 6] are shown in [Table 6]. Note that *GoldCustomer* is the most important entity type due to the definition of the method, which allows inheritance of a fragment of the importance of *Customer*.

e	$\|par_{inh}$ $(e)\|$	$\|chi_{inh}$ $(e)\|$	$\|attr_{inh}$ $(e)\|$	$\|assoc_{inh}$ $(e)\|$	$\|rconn_{inh}$ $(e)\|$	$\boldsymbol{I_{TIM}(e)}$	$\boldsymbol{I^{+}_{TIM}(e)}$
CreditCard	0	0	1	1	6	**2**	**8**
Customer	1	1	3	2	10	**7**	**17**
GoldCustomer	2	0	4	2	14	**8**	**22**
Person	0	3	1	0	0	**4**	**4**
Product	0	0	3	2	14	**5**	**19**
Supplier	1	0	2	1	4	**4**	**8**

Table 6: Results for TIM and TIM+ applied to example of [Fig. 6].

3.5 EntityRank

The EntityRank method [Tzitzikas and Hainaut (2005), Tzitzikas et al. (2007)] is based on link analysis following the same approach as Google's PageRank [Brin and Page (1998)]. Roughly, each entity type is viewed as a state and each association between entity types as a bidirectional transition between them.

The importance of an entity type is the probability that a random surfer is at that entity type with random jumps (q component) or by navigation through relationships ($1-q$ component). Therefore, the resulting importance of the entity types correspond to the stationary probabilities of the Markov chain, given by:

$$I_{ER}(e) = \frac{q}{|\mathcal{E}|} + (1-q) \sum_{e' \in conn(e)} \frac{I_{ER}(e')}{|conn(e')|}$$

Such formulation produces a system of equations. Concretely, for the example of [Fig. 6] such system would be as follows:

$$\begin{aligned}
I_{ER}(CreditCard) &= \tfrac{q}{6} + (1-q)\left(\tfrac{I_{ER}(Customer)}{2}\right)\\
I_{ER}(Customer) &= \tfrac{q}{6} + (1-q)\left(I_{ER}(CreditCard) + \tfrac{I_{ER}(Product)}{2})\right)\\
I_{ER}(GoldCustomer) &= \tfrac{q}{6}\\
I_{ER}(Person) &= \tfrac{q}{6}\\
I_{ER}(Product) &= \tfrac{q}{6} + (1-q)\left(\tfrac{I_{ER}(Customer)}{2} + I_{ER}(Supplier)\right)\\
I_{ER}(Supplier) &= \tfrac{q}{6} + (1-q)\left(\tfrac{I_{ER}(Product)}{2}\right)
\end{aligned}$$

It is important to note that in these methods the importance of an entity comes from the importance of the entities connected to it. The importance flows through associations. For the case of *Customer*, its importance according to the EntityRank comes from *CreditCard* and from *Product*. As *Product* is connected with two entity types, a fragment of its importance (a half) goes to each of its connected entities (*Customer* and *Supplier*).

In our extension to EntityRank we add a new component to the formula in order to jump not only to the connected entity types but also to connected ones through the rule links. The definition is now:

$$I^{+}_{ER}(e) = \frac{q}{|\mathcal{E}|} + (1-q)\left(\sum_{e' \in conn(e)} \frac{I^{+}_{ER}(e')}{|conn(e')|} + \sum_{e'' \in rconn(e)} \frac{I^{+}_{ER}(e'')}{|rconn(e'')|}\right)$$

To compute the importance of the entity types we need to solve the equation system. If we fix that $\sum_{e \in \mathcal{E}} I_{ER}(e) = 1$ and $\sum_{e \in \mathcal{E}} I^{+}_{ER}(e) = 1$ the results obtained are shown in [Tab. 7]. We have chosen $q = 0.15$, which is a common value in the literature [Tzitzikas and Hainaut (2005)].

The results for the entity type *GoldCustomer* indicate that for the extended version of EntityRank (ER+) the importance of such entity type has increased ($I_{ER} = 0.04$, whereas $I^{+}_{ER} = 0.11$). This is because the extended version takes into account the links extracted from the schema rules. Here, *GoldCustomer* has an OCL invariant defined in its context with a participation of the entity type

e	$I_{ER}(e)$	$I^{+}_{ER}(e)$
CreditCard	**0.16**	**0.15**
Customer	**0.30**	**0.25**
GoldCustomer	**0.04**	**0.11**
Person	**0.04**	**0.03**
Product	**0.30**	**0.34**
Supplier	**0.16**	**0.12**

Table 7: Results for ER and ER+ applied to example of [Fig. 6].

Product [see Fig. 6] that produces a rule link between *GoldCustomer* and *Product*. Therefore, a portion of the importance of *Product* flows to *GoldCustomer* in ER+ but not in ER.

3.6 BEntityRank

The BEntityRank [Tzitzikas and Hainaut (2005), Tzitzikas et al. (2007)] is a variation of the previous method specifying that the probability of randomly jumping to each entity type is not the same for each entity type, but it depends on the number of its attributes. The higher the number of attributes, the higher the probability to randomly jump to that entity type. That is:

$$I_{BER}(e) = q\frac{attr(e)}{|\mathcal{A}|} + (1-q)\sum_{e' \in conn(e)} \frac{I_{BER}(e')}{|conn(e')|}$$

Similarly than for EntityRank, the formulation produces a system of equations. In this case the system would be as follows:

$$\begin{aligned}
I_{BER}(CreditCard) &= q\tfrac{1}{9} + (1-q)\left(\tfrac{I_{BER}(Customer)}{2}\right)\\
I_{BER}(Customer) &= q\tfrac{2}{9} + (1-q)\left(I_{BER}(CreditCard) + \tfrac{I_{BER}(Product)}{2})\right)\\
I_{BER}(GoldCustomer) &= q\tfrac{1}{9}\\
I_{BER}(Person) &= q\tfrac{1}{9}\\
I_{BER}(Product) &= q\tfrac{3}{9} + (1-q)\left(\tfrac{I_{BER}(Customer)}{2} + I_{BER}(Supplier)\right)\\
I_{BER}(Supplier) &= q\tfrac{1}{9} + (1-q)\left(\tfrac{I_{BER}(Product)}{2}\right)
\end{aligned}$$

Note that if an entity type is not connected to others through associations, its importance only consists of its percentage of attributes multiplied by the coefficient of random jump (q).

Our extension is in the same way as in EntityRank but taking into account the definition of the attributes component of BEntityRank. The definition is:

$$I^{+}_{BER}(e) = q\frac{attr(e)}{|\mathcal{A}|} + (1-q)\left(\sum_{e' \in conn(e)} \frac{I^{+}_{BER}(e')}{|conn(e')|} + \sum_{e'' \in rconn(e)} \frac{I^{+}_{BER}(e'')}{|rconn(e'')|}\right)$$

To compute the importance of the entity types we need to solve the equation system. We fix that $\sum_{e \in \mathcal{E}} I_{BER}(e) = 1$ and $\sum_{e \in \mathcal{E}} I^{+}_{BER}(e) = 1$ then the results obtained are shown in [Tab. 8]. We choose $q = 0.15$, which is a common value in the literature.

e	$I_{BER}(e)$	$I^{+}_{BER}(e)$
CreditCard	**0.15**	**0.15**
Customer	**0.31**	**0.26**
GoldCustomer	**0.02**	**0.1**
Person	**0.02**	**0.02**
Product	**0.33**	**0.36**
Supplier	**0.16**	**0.11**

Table 8: Results for BER and BER+ applied to example of [Fig. 6.]

3.7 CEntityRank

Finally, the method that we call CEntityRank (m_4 in [Tzitzikas et al. (2007)]) follows the same idea as EntityRank and BEntityRank, but including the generalization relationships. Each generalization between ascendants and descendants is viewed as a bidirectional transition, as in the case of associations. Formally:

$$I_{CER}(e) = q_1 \frac{attr(e)}{|\mathcal{A}|} + q_2 \sum_{e' \in gen(e)} \frac{I_{CER}(e')}{|gen(e')|} + (1 - q_1 - q_2) \sum_{e'' \in conn(e)} \frac{I_{CER}(e'')}{|conn(e'')|}$$

The formulation produces a system of equations as follows:

$$\begin{aligned}
I_{CER}(CreditCard) &= q_1 \tfrac{1}{9} + (1 - q_1 - q_2) \left(\tfrac{I_{CER}(Customer)}{2} \right) \\
I_{CER}(Customer) &= q_1 \tfrac{2}{9} + q_2 \left(\tfrac{I_{CER}(Person)}{2} + I_{CER}(GoldCustomer) \right) \\
&\quad + (1 - q_1 - q_2) \left(I_{CER}(CreditCard) + \tfrac{I_{CER}(Product)}{2} \right) \\
I_{CER}(GoldCustomer) &= q_1 \tfrac{1}{9} + q_2 \left(\tfrac{I_{CER}(Customer)}{2} \right) \\
I_{CER}(Person) &= q_1 \tfrac{1}{9} + q_2 \left(\tfrac{I_{CER}(Customer)}{2} + I_{CER}(Supplier) \right) \\
I_{CER}(Product) &= q_1 \tfrac{3}{9} + (1 - q_1 - q_2) \left(\tfrac{I_{CER}(Customer)}{2} + I_{CER}(Supplier) \right) \\
I_{CER}(Supplier) &= q_1 \tfrac{1}{9} + q_2 \left(\tfrac{I_{CER}(Person)}{2} \right) + (1 - q_1 - q_2) \left(\tfrac{I_{CER}(Product)}{2} \right)
\end{aligned}$$

Our extension includes the rule links as bidirectional transitions for the random surfer. The new definition is:

$$\begin{aligned}
I^{+}_{CER}(e) = {} & q_1 \frac{attr(e)}{|\mathcal{A}|} + q_2 \sum_{e' \in gen(e)} \frac{I^{+}_{CER}(e')}{|gen(e')|} \\
& + (1 - q_1 - q_2) \left(\sum_{e'' \in conn(e)} \frac{I^{+}_{CER}(e'')}{|conn(e'')|} + \sum_{e''' \in rconn(e)} \frac{I^{+}_{CER}(e''')}{|rconn(e''')|} \right)
\end{aligned}$$

Similarly than with EntityRank and BEntityRank, we need to solve this equation system. We fix that $\sum_{e \in \mathcal{E}} I_{CER}(e) = 1$ and $\sum_{e \in \mathcal{E}} I^{+}_{CER}(e) = 1$ then the results obtained are shown in [Tab. 9]. We choose $q_1 = 0.1$ and $q_2 = 0.2$, which are good values as indicated in [Tzitzikas et al. (2007)].

e	$I_{CER}(e)$	$I^{+}_{CER}(e)$
CreditCard	**0.13**	**0.12**
Customer	**0.32**	**0.27**
GoldCustomer	**0.05**	**0.12**
Person	**0.08**	**0.06**
Product	**0.27**	**0.31**
Supplier	**0.16**	**0.12**

Table 9: Results for CER and CER+ applied to example of [Fig. 6.]

4 Experimental Evaluation

We have implemented the seven methods described in the previous section, both the original and the extended versions. We have then evaluated the methods using three distinct case studies: osCommerce [Tort and Olivé (2007)], the UML metaschema [Bauerdick et al. (2004), Bauerdick et al. (2004b)], and EU-Rent [Business Rules Group (2000), Frias et al. (2003)]. The original methods have been evaluated with the input knowledge they are able to process: entity types, attributes, associations, and generalization/specialization relationships of the structural schemas.

	osCommerce	UML Metaschema	EU-Rent schema
Entity Types	346	293	185
Event Types	261	-	120
Attributes	458	93	85
Associations	183	377	152
General Constraints and Derivation Rules	204	170	302
Pre- and Post conditions	220	-	166

Table 10: Schema contents of the case studies.

For osCommerce and EU-Rent, the extended methods have been evaluated with the complete structural schema and the complete behavioural schema (including event types and their pre/post conditions). The schema of osCommerce comprises 346 entity types (of which 261 are event types), 458 attributes, 183 associations, 204 general constraints and derivation rules, and 220 pre- and post conditions. The schema of EU-Rent contains 185 entity types (of which 120 are event types), 85 attributes, 152 associations, 302 general constraints and derivation rules, and 166 pre- and post conditions.

For the UML metaschema there is no behavioral schema and therefore we have only used the complete structural schema. The version of the UML metaschema we have used comprises 293 entity types, 93 attributes, 377 associations, 54 derivation rules, and 116 general constraints. [Tab. 10] summarizes the characteristics of the three case studies. The OCL constraints corresponding to the cardinalities, taxonomies, and association classes are not included in the information of [Tab. 10].

In case of two or more entity types get the same importance, our implementation is non-deterministic: it might rank first any of those. Some enhancements can be done to try to avoid ranking equally-important entity types in a random manner, like prioritizing those with a higher amount of attributes or relationships (or any other measure) in case of ties. However, this does not have an impact to our experimentation.

In the following, we summarize the two main conclusions we have drawn from the study of the result data.

4.1 Correlation Between the Original and the Extended Versions

We study the correlation between the original and the extended version for the previously described methods. Our research aims to know which methods give similar results in both versions.

[Fig. 7] shows, for each method, the results obtained in the original and the extended versions for the osCommerce. The horizontal axis has a point for each of the 85 entity types of the structural schema, ordered descendently by their importance in the original version. The vertical axis shows the importance computed in both versions. The importance has been normalized such that the sum of the importances of all entity types in each method is 100.

As shown in [Fig. 7(g)] the highest correlation between the results of both versions is for the CEntityRank (r=0.931), closely followed by the BEntityRank (r=0.929). The lowest correlation is for the Weighted Simple Method (r=0.61). Similar results are obtained for the UML metamodel. In this case the correlation between the two versions of the Weighted Simple Method is 0.84 and that of the CEntityRank is 0.96. [Tab. 11] summarizes the correlation results for the three case studies.

Our conclusion from this result is that the methods that produce more similar results in both versions are the BEntityRank and the CEntityRank. If this conclusion were confirmed by further experiments, the practical implication would be that, if we only have the fragment of the structural schema comprising the entity and relationship types, their attributes, and the specialization/generalization relationships, then CEntityRank and BEntityRank are the methods of choice.

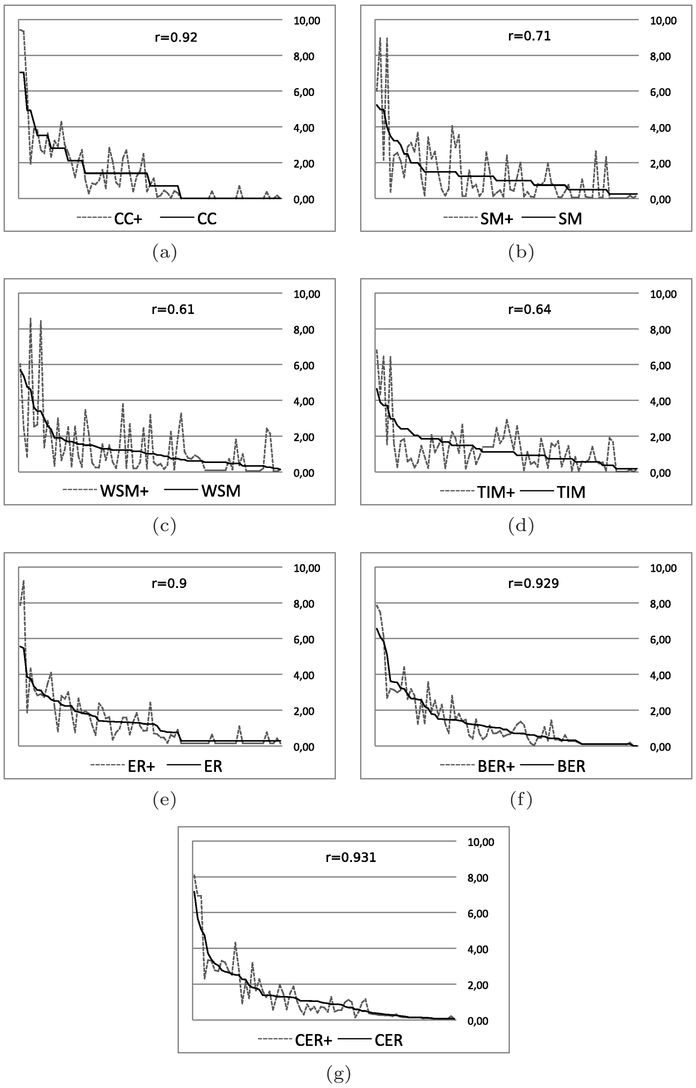


Figure 7: Comparison between base and extended methods applied to the osCommerce.

The reason is that using only those elements, the BEntityRank and CEntityRank methods give results more similar to those that would be obtained taking into account the more of the schema. That is, the more knowledge we take into account the better, but if we only have that fragment of the structural schema then the methods of choice are the CEntityRank and BEntityRank.

On the other hand, the methods based on link analysis (ER, BER and CER) are more constant than those based on occurrence counting (CC, SM, WSM and

Methods	osCommerce	UML Metaschema	EU-Rent
CC - CC+	0.92	0.87	0.93
SM - SM+	0.71	0.85	0.89
WSM - WSM+	0.61	0.84	0.77
TIM - TIM+	0.64	0.94	0.71
ER - ER+	0.9	0.94	0.91
BER - BER+	0.93	0.95	0.94
CER - CER+	0.93	0.96	0.93

Table 11: Correlation coefficients between original and extended methods.

TIM). The main reason for this behavior is the recursive definition of its formulas. The link analysis methods need the importance of other entity types in order to compute the importance of an entity type. This dependency implies an iterative computation to achieve the convergence to a state where the importance flows are in equilibrium. In the case of occurrence counting methods, the computation of the relevance for an entity is totally independent from the other entity types.

This conclusion contrasts with the results reported in the previous work of [Tzitzikas et al. (2007)], which, based on subjective evaluations given by evaluators, concludes that the method that gives the best results is the Simple Method. However, [Fig. 7(b)] shows that the result given by that method considerably changes when more schema knowledge is taken into account.

4.2 Variability of the Original and the Extended Versions

The second experimentation consists of the study of the correlation between original and extended versions separately. Our aim is to know whether it is possible to compare the results of the importance methods and to search for a common behaviour according to if the methods take into account the complete conceptual schema (extended version of methods) or only (a fragment of) the structural schema (original methods).

[Tab. 12], [Tab. 13] and [Tab. 14] show the correlation between each pair of methods (separately, originals and extended versions), in all case studies. It can be seen that, if we exclude the Transitive Inheritance Method (TIM) because it gives the worst results, the correlation in the original versions of the methods ranges from 0.59 to 0.99, while in the extended versions the range is from 0.81 to 0.99.

The conclusion from this result is that the extended versions of the methods, excluding TIM, produce remarkably similar results, which does not happen in the original version. That is, the results obtained by extended versions have a lower degree of variance.

This conclusion is also significant because it assures that the use of the Simple Method (extended version) whose computational cost is very low, and on the

	I_{SM}	I_{WSM}	I_{TIM}	I_{ER}	I_{BER}	I_{CER}
I_{CC}	0.87	0.79	0.06	0.96	0.85	0.86
I_{SM}		0.98	0.15	0.82	0.79	0.92
I_{WSM}			0.16	0.73	0.77	0.90
I_{TIM}				0.06	0.07	0.11
I_{ER}					0.82	0.83
I_{BER}						0.91

	I^+_{SM}	I^+_{WSM}	I^+_{TIM}	I^+_{ER}	I^+_{BER}	I^+_{CER}
I^+_{CC}	0.99	0.97	0.23	0.93	0.82	0.81
I^+_{SM}		0.99	0.26	0.93	0.83	0.86
I^+_{WSM}			0.27	0.91	0.85	0.89
I^+_{TIM}				0.25	0.24	0.30
I^+_{ER}					0.84	0.84
I^+_{BER}						0.91

Table 12: Correlation coefficients between results of original and extended methods for the UML metaschema.

	I_{SM}	I_{WSM}	I_{TIM}	I_{ER}	I_{BER}	I_{CER}
I_{CC}	0.76	0.61	0.43	0.98	0.94	0.94
I_{SM}		0.97	0.79	0.74	0.87	0.88
I_{WSM}			0.79	0.59	0.78	0.76
I_{TIM}				0.40	0.54	0.61
I_{ER}					0.94	0.94
I_{BER}						0.97

	I^+_{SM}	I^+_{WSM}	I^+_{TIM}	I^+_{ER}	I^+_{BER}	I^+_{CER}
I^+_{CC}	0.99	0.99	0.78	0.98	0.92	0.92
I^+_{SM}		0.99	0.79	0.98	0.93	0.93
I^+_{WSM}			0.79	0.98	0.94	0.94
I^+_{TIM}				0.78	0.73	0.83
I^+_{ER}					0.94	0.93
I^+_{BER}						0.97

Table 13: Correlation coefficients between results of original and extended methods for the osCommerce.

	I_{SM}	I_{WSM}	I_{TIM}	I_{ER}	I_{BER}	I_{CER}
I_{CC}	0.88	0.71	0.06	0.99	0.97	0.97
I_{SM}		0.95	0.24	0.87	0.92	0.95
I_{WSM}			0.24	0.71	0.81	0.84
I_{TIM}				0.05	0.06	0.13
I_{ER}					0.97	0.97
I_{BER}						0.99

	I^+_{SM}	I^+_{WSM}	I^+_{TIM}	I^+_{ER}	I^+_{BER}	I^+_{CER}
I^+_{CC}	0.99	0.98	0.65	0.99	0.97	0.95
I^+_{SM}		0.99	0.66	0.99	0.97	0.98
I^+_{WSM}			0.66	0.97	0.96	0.98
I^+_{TIM}				0.66	0.61	0.69
I^+_{ER}					0.98	0.96
I^+_{BER}						0.96

Table 14: Correlation coefficients between results of original and extended methods for the EU-Rent.

other hand it allows the incremental recalculation of the importance of entity types when the schema changes, produces “good-enough” results.

5 Conclusions and Further Work

The visualization and the understanding of large conceptual schemas require the use of specific methods. These methods generate indexed, clustered, summarized or focused schemas that are easier to visualize and understand. Almost all of these methods require computing the importance of each entity type in the schema. We have argued that the objective importance of an entity type in a schema should be related to the amount of knowledge that the schema defines about it. There are several proposals of metrics for entity type importance. All of them are mainly based on the amount of knowledge defined in the schema, but -surprisingly- they only take into account the fragment of that knowledge consisting on the number of attributes, associations and specialization/generalization relationships. A complete conceptual schema also includes cardinalities, general constraints, derivation rules and the specification of events, all of which contribute to the knowledge of entity types.

We have analyzed the influence of that additional knowledge on a representative set of seven existing metrics. We have developed extended versions of each of those metrics. We have evaluated both versions of those methods in three large real-world schemas. The two main conclusions are: (1) Among the original versions of the methods, the methods of choice are those based on the link analysis following the same approach as Google's PageRank; and (2) The extended versions of most methods produce remarkably similar results, which does not happen in the original version.

We plan to continue this work in two main directions. The first, is the experimentation with other large industrial schemas to check whether the above conclusions have a larger experimental basis. The second, is the extension of the work to other existing metrics.

Acknowledgements

Thanks to the people of the GMC group for their useful comments to previous drafts of this paper. We sincerely thank the anonymous referees for their valuable suggestions, which have improved the paper. This work has been partly supported by the Ministerio de Ciencia y Tecnologia and FEDER under project TIN2008-00444/TIN, Grupo Consolidado, and by Universitat Politècnica de Catalunya under FPI-UPC program.

References

[Baroni (2002)] A. L. Baroni.: "Formal definition of object-oriented design metrics". Master's thesis, Vrije Universiteit Brussel, 2002.

[Bauerdick et al. (2004)] H. Bauerdick, M. Gogolla, and F. Gutsche.: "UML 2.0 Validation Results in Form of an EXCEL, PDF, and USE File. University of Bremen". 2004. URL `ftp://ftp.informatik.uni-bremen.de/local/db/papers/uml2004/ocl_uml2.[xls|pdf|use]`.

[Bauerdick et al. (2004b)] H. Bauerdick, M. Gogolla, and F. Gutsche.: "Detecting OCL Traps in the UML 2.0 Superstructure: An Experience Report". In T. Baar, A. Strohmeier, A. M. D. Moreira, and S. J. Mellor, editors, *UML*, volume 3273 of *LNCS*, pages 188–196. Springer, 2004. ISBN 3-540-23307-5.

[Brin and Page (1998)] S. Brin and L. Page.: "The anatomy of a large-scale hypertextual web search engine". In *Computer Networks and ISDN Systems*, pages 107–117. Elsevier Science Publishers B. V., 1998.

[Business Rules Group (2000)] Business Rules Group.: "Defining Business Rules–What Are They Really?". Final Report, July 2000. URL `http://www.businessrulesgroup.org/first_paper/br01c0.htm`

[Castano et al. (1998)] S. Castano, V. D. Antonellis, M. G. Fugini, and B. Pernici.: "Conceptual schema analysis: Techniques and applications". *ACM Trans. Database Syst.*, 23(3):286–332, 1998.

[Frias et al. (2003)] Frias L., Queralt A. and Olivé A.: "EU-Rent Car Rentals Specification". LSI-03-59-R Research Report. 2003. URL `http://www.lsi.upc.edu/dept/techrepts`

[Lindland et al. (1994)] O. I. Lindland, G. Sindre, and A. Sølvberg.: "Understanding quality in conceptual modeling". *IEEE Software*, 11(2):42–49, 1994.

[Moody and Flitman (1999)] D. L. Moody and A. Flitman.: "A Methodology for Clustering Entity Relationship Models – A Human Information Processing Approach". In J. Akoka, M. Bouzeghoub, I. Comyn-Wattiau, and E. Métais editors, *ER 1999*, volume 1728 of *LNCS*, pages 114–130. Springer, 1999. ISBN 3-540-66686-9.

[OMG (2006)] "Object Constraint Language (OCL), version 2.0". Object Management Group (OMG), May 2006. URL `http://www.omg.org/spec/OCL/2.0/`.

[OMG (2009)] "Unified Modeling Language (UML) Superstructure Specification, version 2.2". Object Management Group (OMG), February 2009. URL `http://www.omg.org/spec/UML/2.2/`.

[Olivé and Raventós (2006)] A. Olivé and R. Raventós.: "Modeling events as entities in object-oriented conceptual modeling languages". In Data & Knowledge Engineering, volume 58, number 3, pages 243–262. Elsevier, 2006.

[Olivé (2007)] A. Olivé.: "Conceptual Modeling of Information Systems". Springer-Verlag, 2007. ISBN 3540393897.

[Olivé and Cabot (2007)] A. Olivé and J. Cabot.: "A research agenda for conceptual schema-centric development". In S. B. John Krogstie, Andreas Lothe Opdahl, editor, *Conceptual Modelling in Information Systems Engineering*, pages 319–334. Springer Verlag, 2007.

[Salton (1989)] G. Salton.: "Automatic text processing: the transformation, analysis, and retrieval of information by computer". Addison-Wesley Longman Publishing Co., 1989. ISBN 0201122278.

[Snoeck and Dedene (1998)] M. Snoeck and G. Dedene.: "Existence Dependency: The key to semantic integrity between structural and behavioral aspects of object types". IEEE Transactions on Software Engineering, volume 24, number 4, pages 233–251, 1998.

[Tort and Olivé (2007)] A. Tort and A. Olivé.: "The osCommerce Conceptual Schema". Universitat Politècnica de Catalunya. URL `http://guifre.lsi.upc.edu/OSCommerce.pdf`, 2007.

[Tzitzikas and Hainaut (2005)] Y. Tzitzikas and J.-L. Hainaut.: "How to tame a very large ER diagram (using link analysis and force-directed drawing algorithms)". In L. M. L. Delcambre, C. Kop, H. C. Mayr, J. Mylopoulos, and O. Pastor, editors, *ER 2005*, volume 3716 of *LNCS*, pages 144–159. Springer, 2005. ISBN 3-540-29389-2. URL `http://dx.doi.org/10.1007/11568322_10`.

[Tzitzikas et al. (2007)] Y. Tzitzikas, D. Kotzinos, and Y. Theoharis.: "On ranking rdf schema elements (and its application in visualization)". *J. UCS*, 13(12):1854–1880, 2007.

[Villegas (2009)] A. Villegas.: "Computing the Importance of Entity Types Taking into Account the Whole Schema". Master's thesis, Universitat Politècnica de Catalunya, 2009. URL `http://www.essi.upc.edu/~avillegas/MasterThesis-AVillegas.pdf`

[Villegas and Olivé (2009)] A. Villegas and A. Olivé.: "On Computing the Importance of Entity Types in Large Conceptual Schemas". In C. A. Heuser and G. Pernul, editors, *ER Workshops*, volume 5833 of *LNCS*, pages 22–32. Springer, 2009.

[Yu and Jagadish (2006)] C. Yu and H. V. Jagadish.: "Schema summarization". In U. Dayal, K.-Y. Whang, D. B. Lomet, G. Alonso, G. M. Lohman, M. L. Kersten, S. K. Cha, and Y.-K. Kim, editors, *VLDB*, pages 319–330. ACM, 2006. ISBN 1-59593-385-9.