A Method for Filtering Large Conceptual Schemas

Antonio Villegas and Antoni Olivé

Services and Information Systems Engineering Department
Universitat Politècnica de Catalunya
Barcelona, Spain
{avillegas,olive}@essi.upc.edu

Abstract. We focus on the problem of filtering a fragment of the knowledge contained in a large conceptual schema. The problem appears in many information systems development activities in which people need to operate with a piece of the knowledge contained in that schema. We propose a new method in which a user focuses on one or more entity types of interest for her task at hand, and the method automatically filters the schema in order to obtain a set of entity and relationship types (and other knowledge) relevant to that task, taking into account the interest of each entity type with respect to the focus, computed from the measures of importance and closeness of entity types. The method has been implemented in a prototype tool, and it has been experimented with the schema of the osCommerce and the ResearchCyc ontology.

Keywords: Large Schemas, Filtering, Entity Types, Importance.

1 Introduction

A conceptual schema defines the general knowledge about the domain that an information system needs to know to perform its functions [1]. The conceptual schema of many real-world information systems and the ontologies of broad or general domains are too large to be easily managed or understood.

There are many information system development activities in which people needs to get a piece of the knowledge contained in the conceptual schema. For example, a conceptual modeler needs to check with a domain expert that the knowledge is correct, a database designer needs to implement that knowledge into a relational database, a software tester needs to write tests checking that the knowledge has been correctly implemented in the system components, or a member of the maintenance team needs to change that knowledge.

The largeness of conceptual schemas makes it difficult for a user to get the knowledge of interest to her. This task needs computer support. It was recognised already in the quality framework proposed in [2], and in the study about the usability of ontologies in [3]. The ideal would be an interactive tool in which the user specifies one or more elements of interest and the tool automatically provides a (small) subset of the knowledge contained in the conceptual schema that is

J. Parsons et al. (Eds.): ER 2010, LNCS 6412, pp. 247–260, 2010.

likely to be relevant to the user. The user may then start another interaction with different elements, until she has obtained all knowledge of interest.

There are several techniques and associated tools for the visualization and comprehension of large conceptual schemas or ontologies (see [4,5,6,7,8,9,10,11]). The group of techniques we see more appropriate for our purposes is the one called focus+context. In this techniques, the user focuses on a single element, and the rest of the elements are presented around it, reduced in size until they reach a point that they are no longer visible. However, the techniques do not distinguish between the elements presented around the central one: all are assumed to be equally important to the user. We believe that this assumption is not valid in the above mentioned activities, because not all entity and relationship types are equally important in a large conceptual schema.

Our novel contribution proposes a new method to improve the usability of these schemas, in which a user focuses on one or more entity types of interest for her task, and the method filters the conceptual schema in order to obtain a set of entity and relationship types (and other knowledge) relevant to that task, taking into account the importance of each entity type in the whole schema, and its closeness to the entity types in the user focus. The method has been implemented in a prototype tool, built on top of the USE environment [12]. The tool has been experimented with the conceptual schema of the osCommerce [13], the ResearchCyc ontology [14] and the HL7 information models [15].

The rest of the paper is structured as follows. Next section formalizes the components of conceptual schemas. Section 3 presents an overview of our filtering method. Section 4 defines the representation of the user request to the method. The measures used to filter are defined in Section 5. Section 6 shows the details about the filtered conceptual schemas our method constructs. Finally, Section 7 presents the experimentation with two large schemas and Section 8 concludes the paper and presents future work.

2 Conceptual Schema

In this paper, we deal only with the structural part of the schema, which consists of the elements summarized in Def. 1.

Definition 1. *(Conceptual Schema) A conceptual schema $\mathcal{CS}$ is defined as a tuple $\mathcal{CS} = \langle \mathcal{E}, \mathcal{R}, \mathcal{I}, \mathcal{C}, \mathcal{D} \rangle$, where:*

- *$\mathcal{E}$ is a set of* entity types*. Each $e \in \mathcal{E}$ can contain attributes.*
- *$\mathcal{R}$ is a set of* relationship types *between entity types of $\mathcal{E}$. If $e, e' \in \mathcal{E}$ are participant entity types in a relationship type $r \in \mathcal{R}$ we write $e' \leftrightarrow e$ to indicate that e' and e are directly connected through r.*
- *$\mathcal{I}$ is a set of* IsA *relationships between entity types of $\mathcal{E}$. We write e'* IsA *e $(e, e' \in \mathcal{E})$ to indicate that e' is a direct specialization of e. For example,* Bicycle IsA Vehicle*.*
- *$\mathcal{C}$ is a set of* integrity constraints*.*
- *$\mathcal{D}$ is a set of* derivation rules*.*

We assume that schemas are written in UML/OCL, although the method presented here could be used with schemas written in other languages. The integrity constraints and derivation rules included in $\mathcal{C}$ and $\mathcal{D}$ with no graphical representation in UML are expressed using the OCL language.

3 Filtering Method Overview

This section presents a brief summary of the method we propose to filter large conceptual schemas. The main idea is to extract a reduced and self-contained view from the large schema, that is, a filtered conceptual schema with the knowledge of interest to the user. Figure 1 presents the main tasks of our method.

The method starts with a user dealing with a large conceptual schema. The manual extraction of the knowledge contained in the schema is very difficult and, consequently, a filtering method is needed. The main point here is to adequately capture the information need of the user with respect to the schema to make up a processable knowledge request. The details of this task are presented in Section 4.

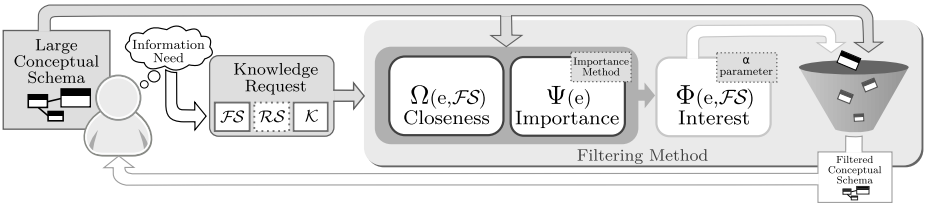


Fig. 1. Method Overview

The method we propose consists in computing some measures taking into account the knowledge of the schema and, specially, the entity types. Our method computes the *interest* as a combination of the *closeness* and *importance* to obtain a ranking of the more interesting (according to the knowledge request) entity types to the user. Section 5 presents these measures.

Our filtering method automatically builds a filtered conceptual schema from a set of interesting entity types and the original schema. Such a filtered conceptual schema is a subset of the original one, and because of its reduced size it is more comprehensible to the user. The details are presented in Section 6. It is important to note that our filtering method only requires user intervention in the definition of the knowledge request. The method uses some elements with default values that can be modified by experimented users.

4 Representing the User Request

The user information need has to be formally defined in order to clarify what the user wants and to make it processable by a computer. We assume that the

information need of a user looking for (a subset of) the knowledge represented in a large conceptual schema includes three components:

- **Focus Set** What the user is interested in about the schema.
- **Rejection Set** What the user is not interested in about the schema.
- **Filter Size** How much knowledge the user wants to obtain from the schema at a given moment.

4.1 Focus Set ($\mathcal{FS}$)

The first component of the user information need is represented by the definition of a focus set.

Definition 2. (Focus Set). *A* focus set $\mathcal{FS}$ *of a conceptual schema* $\mathcal{CS}$ *is a non-empty set of entity types* $e \in \mathcal{E}$.

The focus set includes the entity types the user wants to focus on and works as the conceptual schema viewpoint of the user. Therefore, a focus set is an initial point that should be extended with more knowledge. For example, if the user wants to know information about taxes in the osCommerce schema and she only knows that there exist the concepts *TaxRate* and *TaxClass*, the user could define the focus set $\mathcal{FS} = \{\mathit{TaxRate}, \mathit{TaxClass}\}$.

4.2 Rejection Set ($\mathcal{RS}$)

The second component of the user information need is represented by the definition of a rejection set into the knowledge request.

Definition 3. (Rejection Set). *A* rejection set $\mathcal{RS}$ *of a conceptual schema* $\mathcal{CS}$ *is a set of entity types* $e \in \mathcal{E}$. *The entity types of* $\mathcal{RS}$ *are different from those in* $\mathcal{FS}$ *(*$\mathcal{RS} \cap \mathcal{FS} = \emptyset$*). The default value for* $\mathcal{RS}$ *is the empty set (*$\mathcal{RS} = \emptyset$*).*

The rejection set specifies the entity types the user denotes as not interesting for her knowledge request. Our filtering method ignores those entity types and will not provide knowledge about them to the user.

The osCommerce is a multilingual e-commerce system. Its defined conceptual schema [13] contains the entity type *Language* which is connected with most of the other entity types. Therefore, a user that knows this aspect and does not want to obtain information about *Language* can explicitly define a rejection set $\mathcal{RS} = \{\mathit{Language}\}$.

4.3 Filter Size ($\mathcal{K}$)

The third component of the user information need is represented by the definition of a value to indicate how much knowledge the user wants to obtain from the original schema into the filtered conceptual schema our method returns.

Definition 4. (Filter Size). *The filter size is a natural value* $\mathcal{K}$ *such that* $|\mathcal{FS}| \leq \mathcal{K} \leq |\mathcal{E}|$*, to constraint the number of entity types the user wants to obtain from the schema.*

As an example, a user that defines a focus set with two entity types and a filter size $\mathcal{K} = 10$ will obtain information about the entity types of the focus set and about eight more related entity types.

5 Filtering Measures

In this section we present the measures our filtering method uses to filter the entity types of a conceptual schema.

5.1 Importance of Entity Types (Ψ)

Our filtering method is based on the concept of entity type importance. The importance $\Psi(e)$ of an entity type $e \in \mathcal{E}$ of a conceptual schema $\mathcal{CS}$ is a real number that measures the relative importance of e in $\mathcal{CS}$.

There exist different kinds of methods to compute the importance of entity types in the literature [16]. The simplest family of methods is that based on *occurrence counting* [17,18,19], where the importance of an entity type is equal to the number of characteristics it has in the schema. Therefore, the more characterstics about an entity type, the more important it will be.

Another family of methods are those based in *link analysis* [20,19], where the importance of an entity type is defined as a combination of the importance of the entity types that are connected to it with associations and/or *IsA* relationships. Therefore, the more important the entity types connected to an entity type are, the more important such entity type will be. In these methods the importance is shared through connections, changing from an entity type centered philosophy to a more interconnected approach of the importance.

Table 1 shows the 10 most important entity types of the osCommerce conceptual schema computed by the *CEntityRank* link-analysis method and the occurrence counting *SimpleMethod* [16], and normalized in the range $[0, 1]$. These methods take into account the knowledge of the schema about the entity types $e \in \mathcal{E}$ (including their attributes), the *IsA* relationships $\mathcal{I}$ between them, the relationship types $\mathcal{R}$ and their multiplicities, and the OCL invariants ($\mathcal{C}$ and $\mathcal{D}$).

Finally, there are some methods that also use the information about the existing instances of the entity and relationship types of the conceptual schema. The problem with this family of *instance-dependent* methods [21,22] is that without instances the method is useless.

Our filtering method can be used in connection with any of the existing importance-computing methods. The default importance method used by our filtering method is the above mentioned CEntityRank.

Table 1. Top-10 more important entity types of the osCommerce

(a) CEntityRank

Rank	EntityType e	Importance $\Psi(e)$
1	Language	1
2	Product	0.84
3	Customer	0.62
4	TaxZone	0.57
5	OrderStatus	0.52
6	Order	0.41
7	Currency	0.4
8	TaxClass	0.37
9	Country	0.37
10	Zone	0.35

(b) SimpleMethod

Rank	EntityType e	Importance $\Psi(e)$
1	Product	1
2	Language	0.85
3	Order	0.78
4	Store	0.62
5	Customer	0.61
6	Zone	0.47
7	OrderLine	0.46
8	Attribute	0.46
9	Address	0.42
10	TaxZone	0.41

5.2 Closeness between Entity Types (Ω)

The second measure our filtering method uses is the closeness between entity types. Concretely, the closeness between each *candidate* entity type in the schema and the focus set $\mathcal{FS}$, denoted by $\Omega(e, \mathcal{FS})$. We say that e is a *candidate* entity type if $e \notin \mathcal{FS} \cup \mathcal{RS}$.

There may be several ways to compute the closeness $\Omega(e, \mathcal{FS})$ of a candidate entity type e with respect to the entity types of $\mathcal{FS}$. Intuitively, the closeness of e should be directly related to the inverse of the distance of e to the focus set $\mathcal{FS}$. For this reason, we define:

$$\Omega(e, \mathcal{FS}) = \frac{|\mathcal{FS}|}{\sum_{e' \in \mathcal{FS}} d(e, e')} \tag{1}$$

where $|\mathcal{FS}|$ is the number of entity types of $\mathcal{FS}$ and $d(e, e')$ is the minimum distance between a *candidate* entity type e and an entity type e' belonging to the focus set $\mathcal{FS}$. Intuitively, those entity types that are closer to more entity types of $\mathcal{FS}$ will have a greater closeness $\Omega(e, \mathcal{FS})$.

We assume that a pair of entity types e, e' are directly connected to each other if there is a direct relationship ($e \leftrightarrow e'$) between them or if one entity type is a direct specialization of the other (e *IsA* e' or e' *IsA* e). For these cases, $d(e, e') = 1$. Otherwise, when e, e' are not directly connected, $d(e, e')$ is defined as the length of the shortest path between them traversing relationship types and/or ascending/descending through *IsA* relationships. In these cases, $d(e, e') > 1$. Note that $\sum_{e' \in \mathcal{FS}} d(e, e') = |\mathcal{FS}|$ when e is directly connected to all entity types of $\mathcal{FS}$. If e and e' are not connected (because at least one of them does not participate in relationship types nor *IsA* relationships, or both belong to different connected components of the graph denoted by the schema), then we define $d(e, e') = |\mathcal{E}|$.

5.3 Interest of Entity Types (Φ)

The importance metric is useful when a user wants to know which are the most important entity types, but it is of little use when the user is interested in a

specific subset of entity types, independently from their importance. What is needed then is a metric that measures the interest of a candidate entity type e with respect to a focus set $\mathcal{FS}$. This metric should take into account both the absolute importance of e (as explained in Section 5.1) and the closeness measure of e with regard to the entity types in $\mathcal{FS}$. For this reason, we define:

$$\Phi(e, \mathcal{FS}) = \alpha \times \Psi(e) + (1 - \alpha) \times \Omega(e, \mathcal{FS}) \quad (2)$$

where $\Phi(e, \mathcal{FS})$ is the interest of a candidate entity type e with respect to $\mathcal{FS}$, $\Psi(e)$ the importance of e, and $\Omega(e, FS)$ is the closeness of e with respect to $\mathcal{FS}$.

Note that α is a balancing parameter in the range [0,1] to set the preference between closeness and importance for the retrieved knowledge. An $\alpha > 0.5$ benefits importance against closeness while an $\alpha < 0.5$ does the opposite. The default α value is set to 0.5 and can be modified by the user.

The computation of the interest $\Phi(e, \mathcal{FS})$ for candidate entity types returns a ranking which is used by our filtering method to select the $\mathcal{K} - |\mathcal{FS}|$ top candidate entity types. As an example, Table 2 shows the top-8 entity types with a greater value of interest when the user defines $\mathcal{FS} = \{\textit{TaxRate, TaxClass}\}$, $\mathcal{K} = |\mathcal{FS}| + 8 = 10$ and $\alpha = 0.5$ (the rejection set is the default one, $\mathcal{RS} = \emptyset$). Within the top of interest there may be entity types directly connected to all members of the focus set as in the case of *TaxZone* ($\Omega(\textit{TaxZone}, \mathcal{FS}) = 1.0$) but also entity types that are not directly connected to any entity type of $\mathcal{FS}$ (although they are closer/important).

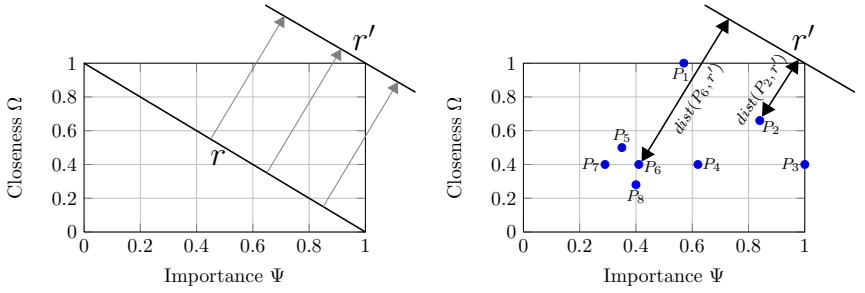
Table 2. Top-8 entity types of interest with regard to $\mathcal{FS} = \{\textit{TaxRate, TaxClass}\}$

Rank	Entity Type (e)	Importance $\Psi(e)$	Distance $d(e, TR)$	Distance $d(e, TC)$	Closeness $\Omega(e, \mathcal{FS})$	Interest $\Phi(e, \mathcal{FS})$
1	TaxZone	0.57	1	1	1.0	0.785
2	Product	0.84	2	1	0.66	0.75
3	Language	1.0	3	2	0.4	0.7
4	Customer	0.62	3	2	0.4	0.51
5	Zone	0.35	2	2	0.5	0.425
6	Order	0.41	3	2	0.4	0.405
7	Special	0.29	3	2	0.4	0.345
8	Currency	0.4	4	3	0.28	0.34

(TR = *TaxRate*, TC = *TaxClass*)

Each candidate entity type e of the conceptual schema $\mathcal{CS}$ can be seen, in a geometrically sense, as a point in a bidimensional space with the axis being the measures of *importance* $\Psi(e)$ and *closeness* $\Omega(e, \mathcal{FS})$. Figure 2(a) shows such bidimensional space with the corresponding axis. Let r be a straight line between the points $(0, \Omega_{max})$ and $(\Psi_{max}, 0)$ of the maximum values of closeness and importance ($\Omega_{max} = \Psi_{max} = 1$ in Fig. 2(a)). We choose r in order to maintain the same proportion between closeness and importance ($\alpha = 0.5$). A straight line r' parallel to r traversing the point $(\Psi_{max}, \Omega_{max})$ indicates the interest line to the user (see Fig. 2(b)).

Taking the importance and the closeness measures for the entity types from Tab. 2, we obtain the coordinates to place them as bidimensional points in the



(a) Movement of r to obtain r'. (b) Distances to the interest indicator r'.

Fig. 2. Geometrical foundation of the concept of Interest of entity types $\Phi(e)$

plane, as shown in Fig. 2(b). The distance between each point in the plane and the straight line r' is inversely proportional to the interest of the entity type the point represents. Figure 2(b) shows that Product placed at point P2=(0.84, 0.66) is of more interest (position 2) than Order (position 6) at point P6=(0.41, 0.4) due to its smaller distance to r'. Note that the balancing parameter α in Eq. 2 can be seen as a modifier of the slope of the straight line r' of Fig. 2(b), in order to prioritize the closeness or importance components. In particular, if we choose $\alpha = 0$ then we only take into account the closeness, and Language (that is at position 3) would be ranked the first.

6 Filtered Conceptual Schema ($\mathcal{F}_{\mathcal{CS}}$)

The main task of our filtering method consists in constructing a filtered conceptual schema, $\mathcal{F}_{\mathcal{CS}}$, from the $\mathcal{K}$ more interesting entity types computed in the previous section, and the knowledge of the original schema (see Fig. 1).

Definition 5. *(Filtered Conceptual Schema) A filtered conceptual schema $\mathcal{F}_{\mathcal{CS}}$ of a conceptual schema $\mathcal{CS} = \langle \mathcal{E}, \mathcal{R}, \mathcal{I}, \mathcal{C}, \mathcal{D} \rangle$ is defined as a tuple $\mathcal{F}_{\mathcal{CS}} = \langle \mathcal{E}_{\mathcal{F}}, \mathcal{R}_{\mathcal{F}}, \mathcal{I}_{\mathcal{F}}, \mathcal{C}_{\mathcal{F}}, \mathcal{D}_{\mathcal{F}} \rangle$, where:*

- *$\mathcal{E}_{\mathcal{F}}$ is a set of entity types filtered from $\mathcal{E}$ of $\mathcal{CS}$ (Section 6.1).*
- *$\mathcal{R}_{\mathcal{F}}$ is a set of relationship types filtered from $\mathcal{R}$ of $\mathcal{CS}$ (Section 6.2).*
- *$\mathcal{I}_{\mathcal{F}}$ is a set of* IsA *relationships filtered from $\mathcal{I}$ of $\mathcal{CS}$ (Section 6.3).*
- *$\mathcal{C}_{\mathcal{F}}$ is a set of integrity constraints filtered from $\mathcal{C}$ of $\mathcal{CS}$ (Section 6.4).*
- *$\mathcal{D}_{\mathcal{F}}$ is a set of derivation rules filtered from $\mathcal{D}$ of $\mathcal{CS}$ (Section 6.5).*

6.1 Filtered Entity Types ($\mathcal{E}_{\mathcal{F}}$)

The entity types $\mathcal{E}_{\mathcal{F}}$ of the filtered conceptual schema $\mathcal{F}_{\mathcal{CS}}$ are those included in the union of the focus set $\mathcal{FS}$, the set $\mathcal{E}_{top}$ of the $\mathcal{K} - |\mathcal{FS}|$ most interesting candidate entity types computed by our method, and the set $\mathcal{E}_{aux}$ of auxiliary entity types due to association projections (see details in Section 6.2).

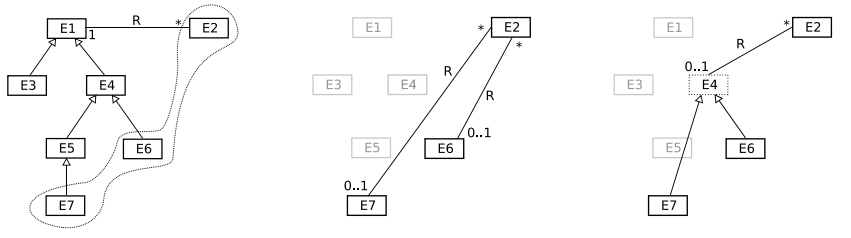
Formally we have $\mathcal{E}_{\mathcal{F}} = \mathcal{FS} \cup \mathcal{E}_{top} \cup \mathcal{E}_{aux}$ and $|\mathcal{E}_{\mathcal{F}}| = \mathcal{K} + |\mathcal{E}_{aux}|$.

6.2 Filtered Relationship Types ($\mathcal{R}_\mathcal{F}$)

The relationship types in $\mathcal{R}_\mathcal{F}$ are those $r \in \mathcal{R}$ whose participant entity types belong to $\mathcal{E}_\mathcal{F}$, or are ascendants of entity types of $\mathcal{E}_\mathcal{F}$ (in which case a projection of r is required). If such relationship types contain an association class, we also include it in $\mathcal{F}_{\mathcal{CS}}$. Formally,

$$\forall r \in \mathcal{R}\ (\forall e \text{ that participates in } r$$
$$(e \in \mathcal{E}_\mathcal{F}\ \vee\ \exists e'\ (e' \text{ is descendant of } e \wedge e' \in \mathcal{E}_\mathcal{F})) \implies r \in \mathcal{R}_\mathcal{F})$$

The projection of a relationship type $r \in \mathcal{R}$ consists in descending the participations of entity types not in $\mathcal{E}_\mathcal{F}$ into each of their descendants in $\mathcal{E}_\mathcal{F}$. Figure 3 shows an example of projection of a relationship type R. The marked area in Fig. 3(a) indicates the entity types that are included in $\mathcal{E}_\mathcal{F}$ (E2, E6 and E7). The relationship type R has two participants. E2 is included in $\mathcal{E}_\mathcal{F}$ while E1 has two indirect descendants, E6 and E7, included in $\mathcal{E}_\mathcal{F}$. Therefore, R should be projected as shown in Fig. 3(b) but, unfortunately, R is repeated, which is correct but increases the complexity of the schema.



(a) Original Relationship (b) Repeated Relationship (c) Projected Relationship

Fig. 3. Result of projecting a relationship to the filtered conceptual schema

There is a special subset of entity types $\mathcal{E}_{aux}$ inside $\mathcal{E}_\mathcal{F}$ that includes the auxiliary entity types that are required to avoid relationship types repetitions. In Fig. 3(a) the closer common ascendant between the entity types in $\mathcal{E}_\mathcal{F}$ (E6 and E7) that descend from the original participant E1 of R is the entity type E4. Therefore, in order to avoid having two R relationship types (connected to E6 and E7, respectively, as shown in Fig. 3(b)) it is necessary to include E4 in $\mathcal{E}_{aux}$, project R to E4, and create *IsA* relationships between the descendants and the auxiliary class (see Fig. 3(c)) to maintain the semantics. It is important to note that if there is only one descendant the auxiliary class is not necessary because the projection of the relationship will be with the descendant itself.

Figure 3(a) shows that the cardinality constraint 1 in E1 has to be changed to 0..1 after the projection of R. This happens because the cardinality constraint of the projected participant E1 must be satisfied for the union of its descendants (E3 and E4), and not for only a subset of them [23].

6.3 Filtered *IsA* Relationships ($\mathcal{I}_\mathcal{F}$)

If e and e' are entity types in $\mathcal{E}_\mathcal{F}$ and there is a direct or indirect *IsA* relationship between them in $\mathcal{I}$ of $\mathcal{CS}$, then such *IsA* relationship must also exist in $\mathcal{I}_\mathcal{F}$ of $\mathcal{F}_{\mathcal{CS}}$. Formally we have $\forall e', e \in \mathcal{E}_\mathcal{F}\ ((e'\ IsA^+\ e) \in I \implies (e'\ IsA^+\ e) \in I_\mathcal{F})$[1].

Figure 4(a) shows a fragment of an original schema where E1, E4, E5 and E6 are the entity types included in $\mathcal{E}_\mathcal{F}$. Figure 4(b) presents the *IsA* relationships included in $\mathcal{I}_\mathcal{F}$ of $\mathcal{F}_{\mathcal{CS}}$. Note that we maintain the direct *IsA* relationships of the original schema between entity types in $\mathcal{E}_\mathcal{F}$, as in the case of E6 *IsA* E5. We also keep the semantics by adding $\mathcal{I}_\mathcal{F}$ of $\mathcal{F}_{\mathcal{CS}}$ the new *IsA* relationships E4 *IsA* E1 and E5 *IsA* E1 as shown in Fig. 4(b).

(a) Original *IsA* Relationships (b) Filtered *IsA* Relationships

Fig. 4. Example of filtering *IsA* relationships

6.4 Filtered Integrity Constraints ($\mathcal{C}_\mathcal{F}$)

The integrity constraints $\mathcal{C}_\mathcal{F}$ included in $\mathcal{F}_{\mathcal{CS}}$ are a subset of the integrity constraints $\mathcal{C}$ of $\mathcal{CS}$. Concretely, the included integrity constraints are those whose expressions only involve entity types from $\mathcal{E}_\mathcal{F}$. Formally we have $\forall\ c \in \mathcal{C}\ (\forall e$ involved in $c\ (e \in \mathcal{E}_\mathcal{F}) \implies c \in \mathcal{C}_\mathcal{F})$

An entity type can be referenced by means of its attributes, its participations in relationship types or by referencing the entity type itself. As an example, the integrity constraint `ic1` in Fig. 5 has A and B as its participants. A is referenced as the context of the constraint and by means of its attribute *a1* in the OCL expression `self.a1`. Also, B is referenced by means of its attribute *b1* in the OCL expression `self.b.b1`. Our method only includes `ic1` into $\mathcal{C}_\mathcal{F}$ of $\mathcal{F}_{\mathcal{CS}}$ if both A and B are entity types in $\mathcal{E}_\mathcal{F}$.

6.5 Filtered Derivation Rules ($\mathcal{D}_\mathcal{F}$)

The derivation rules $\mathcal{D}_\mathcal{F}$ included in $\mathcal{F}_{\mathcal{CS}}$ are those rules $\mathcal{D}$ of $\mathcal{CS}$ whose expressions only involve entity types from $\mathcal{E}_\mathcal{F}$. Formally we have $\forall\ d \in \mathcal{D}\ (\forall e$ involved in $d,\ (e \in \mathcal{E}_\mathcal{F}) \implies d \in \mathcal{D}_\mathcal{F})$

The derivation rule `dr1` in Fig. 5 is included in $\mathcal{D}_\mathcal{F}$ if both A and B are entity types in $\mathcal{E}_\mathcal{F}$. If only $B \in \mathcal{E}_\mathcal{F}$, our method marks the derived attribute *b2* as materialized and does not include `dr1` in $\mathcal{D}_\mathcal{F}$ because that derivation rule also references the entity type A wich is not included in $\mathcal{E}_\mathcal{F}$.

[1] Note that "IsA^+" denotes the transitive closure of *IsA* relationships.

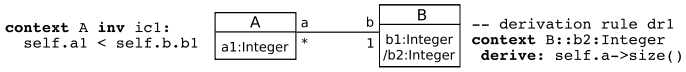


Fig. 5. Example of integrity constraint (`ic1`) and derivation rule (`dr1`)

7 Experimentation

This section presents the results obtained by our filtering method in two real large schemas: the osCommerce [13], and the ResearchCyc (`research.cyc.com`). Table 3 shows some metrics of both conceptual schemas.

Table 3. Conceptual schema characteristics of two large schemas

	Entity Types	Attributes	Relationship Types	*IsA* Relationships
osCommerce	84	209	183	28
ResearchCyc	26,725	1,060	5,514	43,323

7.1 osCommerce

The conceptual schema of the osCommerce [13] includes the elements shown in Tab. 3 and also 204 general constraints and derivation rules. Figure 6 shows the filtered conceptual schema $\mathcal{FCS}$ that results when the user selects $\mathcal{K} = 10$ and wants to know more about $\mathcal{FS} = \{TaxRate, TaxClass\}$.

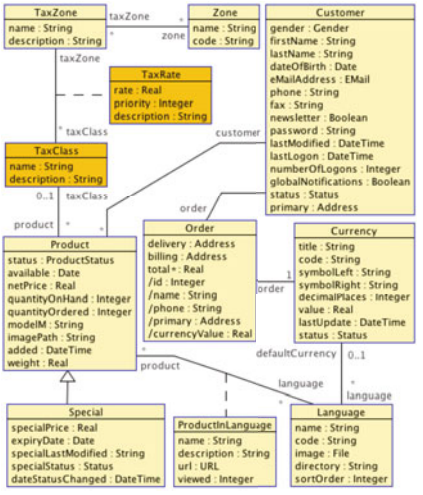
Figure 6(b) presents the integrity constraints and derivation rules ($\mathcal{C}_{\mathcal{F}}$ and $\mathcal{D}_{\mathcal{F}}$) included in $\mathcal{FCS}$. The derivation rules of the derived attributes *id*, *name*, *phone*, *primary* and *currencyValue* of *Order* in Fig. 6(a) are included in $\mathcal{D}_{\mathcal{F}}$ because they only use information contained in $\mathcal{FCS}$. Additionally, we mark each derived attribute that has been materialized with an asterisk ($*$) at the end of its name (as in the case of *total* of *Order*) and its derivation rule is hidden because it uses information about entity types out of $\mathcal{FCS}$.

7.2 ResearchCyc

ResearchCyc knowledge base contains more than 26,000 entity types and is defined using the CycL language [24]. Our experimentation with ResearhCyc has been done with a UML version obtained through a conversion process from CycL.

The anatomy of this ontology has the peculiarity that it contains a small core of abstract concepts strongly connected through high-level relationship types. The rest of the concepts are all descendants of such core. The interesting knowledge the user obtains with the filtered conceptual schema in Fig. 7 about Cancer are the *IsA* relationships with other interesting concepts because the relationship types are defined only between top elements in the hierarchy of concepts.

The experiments we have done with our implementation show that, starting with a focus set of up to three entity types, the time required to compute the interest and filter the ResearchCyc ontology is about half a second (the average of 100 experiments is 0.53 seconds, with a standard deviation of 0.31).



(a) Filtered Schema

```
-- Integrity Constraints

context TaxClass inv nameIsUnique:
    TaxClass.allInstances->isUnique(name)

context TaxZone inv nameIsUnique:
    TaxZone.allInstances->isUnique(name)

context Customer inv eMailIsUnique:
    Customer.allInstances->isUnique(eMailAddress)

context Language inv codeAndNameAreUnique:
    Language.allInstances->isUnique(name) and
    Language.allInstances->isUnique(code)

context Language inv nameIsUnique:
    Language.allInstances->forAll(l |
        l.productInLanguage->isUnique(name))

context Currency inv codeAndTitleAreUnique:
    Currency.allInstances->isUnique(title) and
    Currency.allInstances->isUnique(code)

-- Derivation Rules

context Order::id:Integer derive:
    if Order.allInstances -> size() = 0 then 0
    else Order.allInstances -> sortedBy(id)
            -> last().id() + 1
    endif

context Order::name:String derive:
        self.customer.firstName

context Order::phone:String derive:
        self.customer.phone

context Order::eMail:EMail derive:
        self.customer.eMailAddress

context Order::primary:Address derive:
        self.customer.primary

context Order::currencyValue:Real derive:
        self.currency.value
```

(b) $\mathcal{C}_{\mathcal{F}}$ and $\mathcal{D}_{\mathcal{F}}$

Fig. 6. Filtered conceptual schema for $\mathcal{FS} = \{\text{TaxRate, TaxClass}\}$ in the osCommerce

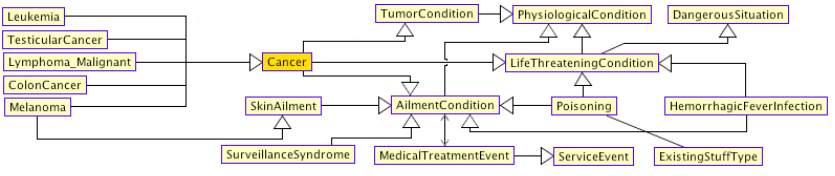


Fig. 7. Filtered conceptual schema for $\mathcal{FS} = \{Cancer\}$ and $\mathcal{K} = 18$ in the ResearchCyc

8 Conclusions and Future Work

We have focused on the problem of filtering a fragment of the knowledge contained in a large conceptual schema. The problem appears in many information systems development activities in which people needs to operate for some purpose with a piece of the knowledge contained in that schema.

We have proposed a filtering method in which a user indicates a focus set consisting of one or more entity types of interest, and the method determines a subset of the elements of the original schema that is likely to be of interest to the user. In order to select this subset, our method measures the interest of each entity type with respect to the focus set based on the importance and closeness.

We have implemented our method in a prototype tool built on top of the USE environment. We have experimented it with two large schemas. In both cases, our tool obtains the filtered schema in a short time. Using our prototype tool it is practical for a user to specify a focus set, to obtain a filtered schema, and to repeat the interaction until the desired knowledge has been obtained.

We plan to improve our method in several ways. One improvement is to take into account the importance of the relationship types. In the current method, we only use the importance of entity types and assume that all relationship types are equally important. This improvement requires the definition of a convenient metric of the importance of relationship types, which does not yet exist. Another enhancement consists in a fine-grained filter of integrity constraints and derivation rules in order to hide only those OCL expressions that reference entity types out of the user focus instead of hiding the whole constraint or rule. Finally, we plan to conduct experiments to precisely determine the usefulness of our method to real users.

Acknowledgements

Thanks to the people of the GMC group for their useful comments to previous drafts of this paper. This work has been partly supported by the Ministerio de Ciencia y Tecnologia under TIN2008-00444 project, Grupo Consolidado, and by Universitat Politècnica de Catalunya under FPI-UPC program.

References

1. Olivé, A.: Conceptual Modeling of Information Systems. Springer, Heidelberg (2007)
2. Lindland, O.I., Sindre, G., Sølvberg, A.: Understanding quality in conceptual modeling. IEEE Software 11(2), 42–49 (1994)
3. Conesa, J., Storey, V.C., Sugumaran, V.: Usability of upper level ontologies: The case of researchcyc. Data & Knowledge Engineering 69(4), 343–356 (2010)
4. Tzitzikas, Y., Hainaut, J.L.: On the visualization of large-sized ontologies. In: AVI 2006, Working Conf. on Advanced Visual Interfaces, pp. 99–102. ACM, New York (2006)
5. Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.: Ontology visualization methods-a survey. ACM Computing Surveys 39(4), 10 (2007)
6. Lanzenberger, M., Sampson, J., Rester, M.: Visualization in ontology tools. In: Intl. Conf. on Complex, Intelligent and Software Intensive Systems, pp. 705–711. IEEE Computer Society, Los Alamitos (2009)
7. Shoval, P., Danoch, R., Balabam, M.: Hierarchical entity-relationship diagrams: the model, method of creation and experimental evaluation. Requirements Engineering 9(4), 217–228 (2004)
8. Rokach, L., Maimon, O.: Clustering methods. In: Data Mining and Knowledge Discovery Handbook, ch. 15, pp. 321–352. Springer, Heidelberg (2005)
9. Campbell, L.J., Halpin, T.A., Proper, H.A.: Conceptual schemas with abstractions making flat conceptual schemas more comprehensible. Data & Knowledge Engineering 20(1), 39–85 (1996)
10. Kuflik, T., Boger, Z., Shoval, P.: Filtering search results using an optimal set of terms identified by an artificial neural network. Information Processing & Management 42(2), 469–483 (2006)
11. Hanani, U., Shapira, B., Shoval, P.: Information filtering: Overview of issues, research and systems. User Modeling and User-Adapted Interaction 11(3), 203–259 (2001)

12. Gogolla, M., Büttner, F., Richters, M.: USE: A UML-based specification environment for validating UML and OCL. Science of Computer Programming (2007)
13. Tort, A., Olivé, A.: The osCommerce Conceptual Schema. Universitat Politècnica de Catalunya (2007), http://guifre.lsi.upc.edu/OSCommerce.pdf
14. Lenat, D.B.: Cyc: a large-scale investment in knowledge infrastructure. Communications of the ACM 38(11), 33–38 (1995)
15. Villegas, A., Olivé, A., Vilalta, J.: Improving the usability of hl7 information models by automatic filtering. In: IEEE 6th World Congress on Services (SERVICES) (2010), http://www.computer.org/portal/web/csdl/doi/10.1109/SERVICES.2010.32
16. Villegas, A., Olivé, A.: On computing the importance of entity types in large conceptual schemas. In: Heuser, C.A., Pernul, G. (eds.) ER 2009 Workshops. LNCS, vol. 5833, pp. 22–32. Springer, Heidelberg (2009)
17. Castano, S., De Antonellis, V., Fugini, M.G., Pernici, B.: Conceptual schema analysis: techniques and applications. ACM Transactions on Database Systems 23(3), 286–333 (1998)
18. Moody, D.L., Flitman, A.: A methodology for clustering entity relationship models-a human information processing approach. In: Akoka, J., Bouzeghoub, M., Comyn-Wattiau, I., Métais, E. (eds.) ER 1999. LNCS, vol. 1728, pp. 114–130. Springer, Heidelberg (1999)
19. Tzitzikas, Y., Kotzinos, D., Theoharis, Y.: On ranking rdf schema elements (and its application in visualization). Journal of Universal Computer Science 13(12), 1854–1880 (2007)
20. Tzitzikas, Y., Hainaut, J.L.: How to tame a very large er diagram (using link analysis and force-directed drawing algorithms). In: Delcambre, L.M.L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, Ó. (eds.) ER 2005. LNCS, vol. 3716, pp. 144–159. Springer, Heidelberg (2005)
21. Yu, C., Jagadish, H.V.: Schema summarization. In: VLDB 2006, 32nd Intl. Conf. on Very Large Data Bases, pp. 319–330 (2006)
22. Yang, X., Procopiuc, C.M., Srivastava, D.: Summarizing relational databases. In: VLDB 2009, 35th Intl. Conf. on Very Large Data Bases, pp. 634–645 (2009)
23. Conesa, J.: Pruning and refactoring ontologies in the development of conceptual schemas of information systems. PhD thesis, UPC (2008)
24. Lenat, D.B., Guha, R.V.: The evolution of cycl, the cyc representation language. ACM SIGART Bulletin 2(3), 84–87 (1991)