A Semi-Automatic Approach for the Integration of Structural Karlstad Enterprise Modeling Schemata

Peter Bellström

Information Systems
Karlstad University
651 88 Karlstad, SWEDEN

ABSTRACT

In this paper, we describe and discuss a semi-automatic approach for the integration of structural Karlstad Enterprise Modeling (EM) Schemata. The focus is on the implementation-independent level and therefore we treat an EM schema as a high-level description of data for some part of the future information system. Our point of departure is the classic four-phase integration process comprised of pre-integration, comparison of the schemata, conforming the schemata and, merging and restructuring. In relation to the semi-automatic approach described and discussed, we argue that several rules and knowledge repositories should be used to facilitate the whole integration process. However, it is also argued that the domain experts are still a very important source of knowledge and should therefore also be involved during the whole integration process. The research approach is inspired by design science in which the end product should be a useful artifact. In this paper, the artifact is provided in the described semi-automatic approach for the integration of Structural Karlstad EM schemata. As its main contribution, the paper offers a holistic view of the described and discussed semi-automatic approach for the integration of structural Karlstad EM schemata.

Keywords: Schema Integration, Semi-Automatic Approach, Karlstad Enterprise Modeling Approach, Implementation-Independent level

INTRODUCTION

Schema integration is the problem of integrating two or more source schemata into one integrated schema. Batini et al. (1986) describe schema integration as “the activity of integrating the schemas of existing or proposed databases into a global, unified schema.” (p. 323). Schema integration is a key issue when developing new information systems, including standalone applications, e-services, and so forth. This is motivated since the new information system is often described and illustrated in a set of schemata with or without textual descriptions. However, each schema illustrates some part of the new information system with a focus on how some domain expert and/or group of domain experts describe their part of the information system often making it into a set of heterogeneous schemata. A schema might also be represented in many forms. Bernstein et al. (2011) emphasize this and describe as schema as “a formal structure that represents an engineered artifact, such as a SQL schema, XML schema, entity-relationship diagram, ontology description, interface definition, or form definition” (p. 695). In this paper, however, we focus on the notation of the Karlstad Enterprise Modeling (EM) approach and therefore treat and describe a schema as a graphical description illustrating the structural (static) part, the data, of an information system being

https://openaccess.cms-conferences.org/#/publications/book/978-1-4951-2091-6

modeled using the EM approach. Schema integration can also be conducted on at least two levels of abstraction: on an *implementation-independent level* closer to humans, and on an *implementation-dependent level* closer to programming languages and computers. In this paper we focus on the former, the *implementation-independent level*.

Automation of the schema integration process has been a dream for many researchers in the field. However, due to its complexity it is not realistic to aim at full automation (Stumptner et al., 2004). The idea of aiming for a semi-automatic approach, however, was addressed in the mid-80s. For instance, in Convent (1986) the author emphasizes that computer applications should be used to assist the domain experts and designers doing schema integration. In our approach we have therefore also chosen the semi-automatic approach, in which not only rules and domain repositories are used but the domain experts are also very much involved during the whole integration process.

As a result, this paper presents a semi-automatic approach for the integration of structural Karlstad EM schemata. Within each phase of the integration process, we address important aspects that need to be taken into consideration in a semi-automatic approach. We emphasize the usage of rules and domain repositories to aid in the schema integration process and also the involvement of domain experts during the whole integration process since they are also very important sources of knowledge.

This paper is structured as follows: in section two we present the Karlstad EM approach focusing on the structural part (static dependencies). In section three, we describe the adopted research approach and in section four the main contribution of this paper: the described and discussed semi-automatic approach for integration of structural Karlstad EM schemata. In section four we also describe related work and distinguish it from our own. In section five, we give a small and illustrative example on how parts of the semi-automatic approach for integration of structural Karlstad EM schemata could be used. Finally, the paper closes with a summary and future work.

THE KARLSTAD ENTERPRISE MODELLING APPROACH

The Karlstad Enterprise Modelling (EM) approach that is adopted in this work refers to a modeling approach developed at Karlstad University, Sweden. The modeling approach has primitives for illustrating the *pragmatic*, the *semantic* (both static and dynamic aspects) and the *syntactic* aspects of the information system that is being modeled. The pragmatic aspects are modeled using a set of pragmatic dependencies illustrating goals, problems and opportunities (Gustas and Gustiené, 2008). Among these goals, problems and opportunities it is also possible to illustrate positive and negative influences. The semantic aspects are modeled using a set of static aspects and a set of dynamic aspects. The dynamic aspects can further be divided into communication dependencies (actors, actions and flows) and state dependencies (states and conditions). The syntactic aspects are modeled using a set of CASE-tools dependent syntactic elements (Gustas and Gustiené, 2004). Nevertheless, in this work we focus on the structural aspects, static dependencies (Figure 1), used to illustrate concepts and dependencies between these.

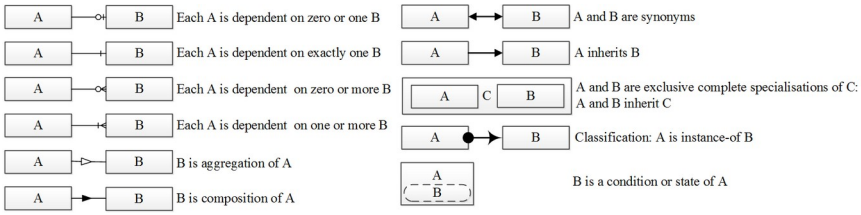

Figure 1. Representation of static dependencies in EM (adapted and modified from Gustas and Gustiené, 2004)

Within the structural aspects, the only primitive in EM that is given a name (a label) are concepts. A concept is drawn as a box and the label is written in it. To illustrate that two concepts are connected to each other through a dependency, a line between them is drawn. Besides that, it is possible to mark cardinality and to specify a part-of dependency (aggregation and composition), that two concepts are synonyms, an is-a dependency (inherits and specialization), an instance-of dependency (classification), and a condition or state.

Finally, it should be noted that in EM we do not distinguish between classes (entities) and properties (attributes) but instead draw them all as concepts (boxes). This is motivated since we here focus on concepts and dependencies

between concepts illustrating an implementation-independent level of the information system being modeled.

RESEARCH APPROACH

The approach adopted in this research is inspired by design science (Hevner, 2007; Hevner and Chatterjee, 2010; Hevner et al., 2004; Iivari, 2007; March and Smith, 1995). In design science research the IT-artifact is always at the center, or as Hevner et al. (2004) puts it: "The result of design-science research in IS is, by definition, a purposeful IT artifact created to address an important organizational problem" (p. 82). Furthermore, in design science the IT-artifact, the research output, might either be a *construct*, a *model*, a *method* or an *instantiation* (Hevner et al., 2004; March and Smith, 1995). March and Smith (1995) describe the four research outputs as follows:

- "*Construct* or concepts form the vocabulary of a domain. They constitute a conceptualization used to describe problems within the domain and to specify their solutions." (p. 256)
- "A *model* is a set of propositions or statements expressing relationships among constructs. In design activities, models represent situations as problem and solution statements." (p. 256)
- "A *method* is a set of steps (an algorithm or guideline) used to perform a task." (p. 257)
- "An instantiation is the realization of an artifact in its environment." (p. 258)

However, to end up with an IT artifact, a research output, it is important to follow some research method and/or research guidelines. In Hevner et al. (2004) the authors describe seven research guidelines that should "assist researchers, reviewers, editors, and readers to understand the requirements for effective design-science research" (p. 82). In this work the seven guidelines have been applied in the following way.

1. *Design as an Artifact*. The proposed semi-automatic approach for the integration of structural Karlstad EM schemata is the research outcome. This means that the proposed approach is the artifact and it can be classified as a method according to the list of research outputs described by March and Smith (1995).

2. *Problem Relevance.* Almost 30 years ago Navathe et al. (1986) stated that schema integration is a complex, time-consuming and error-prone task. This statement is still true. Research conducted in the area has solved many problems related to schema integration. However, since new approaches, methods and modeling languages are developed and old ones change new research problems and questions are stated and tackled all the time. For instance, schema matching, the second phase in the integration process, has many application areas where Bernstein et al. (2011) mention the database field, knowledge-based applications, health care, and web applications as four examples.

3. *Design Evaluation.* Hevner et al. (2004) mention five design evaluation methods: *observational*, *analytical*, *experimental*, *testing* and *descriptive*. In this work we have evaluated the research results using the descriptive method, which includes both informed arguments and scenarios. However, since building a tool, an instantiation (March and Smith, 1995), for our approach is part of future work we do claim that we have conducted all three cycles (relevance cycle, rigor cycle, and design cycle) as described in Hevner (2007).

4. *Research Contribution.* Several delimited studies on how to semi-automate parts of integrating structural Karlstad EM schemata have been reported (e.g. Bellström, 2010b; 2012). However, to our knowledge there is no semi-automatic approach for integrating structural Karlstad EM schemata that deals with all four phases of the integration process as described by Batini et al. (1986).

5. *Research Rigor*. In our approach we combine approaches, methods and strategies to propose an integration process for the Karlstad EM approach. Several of these methods, approaches and strategies have been implemented and tested (e.g. Bellström and Vöhringer, 2011a; 2011b; Vöhringer, 2010) in what March and Smith (1995) describe as an instantiation, a software prototype, while other methods, approaches and strategies are based on rules developed for future implementation (e.g. Bellström, 2010b; 2012).

6. *Design as a Search Process*. Our approach has been developed incrementally and iteratively for some time.

During these iterations we have developed and search for applicable approaches, methods and strategies to all four phases of the integration process as described by Batini et al. (1986). However, since the modeling approach adopted in this work differs from the more classic modeling approaches, such as the ER and the UML, several adaptions to the identified approaches, methods and strategies have been made.

7. *Communication of Research*. In Hevner et al. (2004) the authors state that research output of design science should be communicated to a technology-oriented audience as well as to a management-oriented audience. Since the research results described in this paper should be viewed as a the first step towards a semi-automatic approach for the integration of structural Karlstad EM schemata our usage of the language should appeal to both target groups, meaning that the paper as such is part of fulfilling the last research guideline.

Finally, it should be mentioned that design science research is not a new research approach. It has been adopted and used in many disciplines, such as Computer Science, Software Engineering and Information Systems, for a rather long time now (Iivari, 2007). Examples of recent research that has adopted design science are given in Persson and Stirna (2010), in which the authors used design science to define a competence profile for practitioners of enterprise modeling and in Bellström and Kop (2012), in which we used design science to develop a framework for schema quality in the schema integration process.

A SEMI-AUTOMATIC APPROACH FOR SCHEMA INTEGRATION

In our semi-automatic approach we follow the classic four phase integration process as proposed by Batini et al. (1986) comprised of *pre-integration*, *comparison of the schemata*, *conforming the schemata* and, *merging and restructuring* (Figure 2).

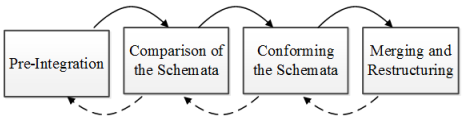

Figure 2. The four phases of the schema Integration process (adapted and modified from Bellström, 2006a).

In the first phase, *pre-integration*, the input schemata are processed and adapted to facilitate the following three phases of the integration process. In the second phase, *comparison of the schemata*, the input schemata are compared with the aim to identify similarities as well as differences and in the third phase, *conforming the schemata*, the identified similarities and differences are resolved. Finally, in the fourth and last phase, *merging and restructuring*, the schemata are first merged into an intermediate schema and after that restructured to fulfill several quality criteria and/or quality factors. At the end of this phase the designers should have a global schema that can be passed to the phases implementing the information system. In the rest of this section we go through each phase of the integration process. In doing so, we both state and argue for the usage of methods, approaches and strategies (e.g. rules and repositories) that can aid in the process of integrating structural Karlstad EM schemata. More specifically this means describing and discussing the research output of this paper: the proposed semi-automatic approach for the integration of structural Karlstad EM schemata. One last remark is required before we continue with each integration phase in the integration process. In developing the proposed semi-automatic approach we have not only been inspired but also influenced by the work first introduced in Bellström and Vöhringer (2009) and later on developed and extended in Bellström and Vöhringer (2011a; 2011b). However, in these publications we focused on developing a language independent integration approach whereas in this work we focus on developing a semi-automatic approach for the integration of structural Karlstad EM schemata, which is a clear and important distinction between the approaches.

Pre-Integration

Pre-integration is the first phase in the integration process and its main purpose is to collect as much information of the input schemata as possible and adjust the input schemata to facilitate the phases that follows. In earlier research within the area of schema integration, this phase has not always been mentioned. For instance, in Batini et al. (1986) only three out of twelve analyzed methods mentioned pre-integration as a phase in the integration process. Also

https://openaccess.cms-conferences.org/#/publications/book/978-1-4951-2091-6

Song (1995) pointed out that pre-integration has often been overlooked. However, over the years the importance of the pre-integration phase and the tasks to perform during it has been emphasized. For instance in Song (1997) the author states that pre-integration is an important phase within the schema integration process.

In pre-integration several tasks should be carried out. In Batini et al. (1986) the authors stated that the integration strategy should be decided. Furthermore, these strategies could be divided into binary strategies including ladder and balanced and n-ary strategies including one-shot and iterative. In Song (1995) the author stated three tasks to be performed in pre-integration: translate all input schemata into the modeling language chosen, recognize and resolve conflicts within each schema and choose the integration strategy. In our approach we follow the work reported in Bellström and Vöhringer (2011b), and in Bellström et al. (2012) also adapted and modified to behavioral schemata, and therefore carry out at least the six tasks addressed below.

(1) Translating the input schemata into the chosen modeling language is applicable if the information system is modeled using different modeling languages. In our case we translate all input schemata into the Karlstad EM approach since that is the modeling language chosen and used in our research. Once again it should be noted that in the presented research we focus on the structural aspects (static dependencies) of the modeled information system and leave out the dynamic aspects for future research. In Bellström and Kop (2012) we addressed semi-automatic schema pre-integration and in the comparison of related work, translation of input schemata was one of the tasks that was mostly addressed, although not always defined as a pre-integration task.

(2) Analyzing the input schemata to adapt concept names is conducted to first of all facilitate comparison of the source schemata. In Bellström et al. (2008) we pointed out that standardization of concept notions was one task to perform to facilitate the following integration phases. Standardization could be carried out through the usage of naming guidelines in which, for instance, concept names are written in singular. Besides this, in Bellström and Vöhringer (2009) it was mentioned that a stemmer (Hull, 1996) could be used to bring a concept name to its base form. To aid in the process of standardization of concept names the approach proposed in Sorrentino et al. (2009) could also be used. In Sorrentino et al. (2009) the authors proposed and described a semi-automatic method that normalizes the schema labels meaning, abbreviations, acronyms, and expands compound terms. Nevertheless, irrespective of method used to adapt concept names, the approach should counteract the occurrence of semantic loss (Bellström, 2009).

(3) Analyzing the input schemata to disambiguate concept names is a task that is conducted to collect information for the domain experts in order to manually verify or decline a suggestion from an integration tool. This task could either be conducted manually by the domain experts or automatically by an ontology (Gruber, 1993) and/or lexicon such as WordNet (Miller, 1995). A more detailed discussion on how to disambiguate concept names is given in Vöhringer (2010).

(4) Analyzing the input schemata aiming to recognize and resolve intra-schema conflicts is a task that is conducted to both recognize and resolve problems within one schema. In doing so, not only homonyms but also synonyms should be recognized and resolved. Within each source schemata we also need to deal with hypernym-hyponym dependencies (is-a) and holonym-meronym dependencies (part-of). For this reason, this task could be viewed as a minor version of phase two and three in the integration process. However, as will be discussed in more depth in conforming the schemata, while resolving conflicts it is important that we do not lose any concept names and/or dependencies since these might have a meaning for one or several domain expert(s). Losing a concept name and/or dependency would in most cases cause semantic loss (Bellström, 2009).

(5) Introducing missing concepts and relationships is a task that is performed if the input schemata include so called many-to-many dependencies (links) and/or if a domain ontology (Gruber, 1993) and/or taxonomy is available. So-called many-to-many dependencies can be translated into two many-to-one dependencies with a new concept placed in between the two. This task can either be performed automatically or manually together with the domain experts. Introduction of missing relationships are performed if and only if a domain ontology and/or taxonomy is available and the purpose is to automatically enrich and add relationships to the schemata prepared for integration.

(6) Selecting the integration strategy is the task in which the integration processing strategy is decided. In our approach we have decided to use the binary ladder strategy (Batini and Lenzerini, 1984; Batini et al., 1986; Batini et al., 1992). This is motivated since binary ladders do not only simplify comparison and conforming the schemata

(Batini et al. (1986) but also ensure that enough intermediate schemata are produced for quality insurance (Bellström and Kop, In press).

Comparison of the Schemata

Comparison of the schemata is the second phase in the integration process. Over the years it has received a lot of attention in the research community. Comparison of the schemata has for instance been mentioned as "The most challenging problem in schema integration" (Johannesson, 1993, p. 19). It has also been called an important (Song, 1995) and difficult (Doan et al., 2004; Ekenberg and Johannesson, 1996; Lee and Ling, 2003) phase of schema integration. Finally, schema matching has also been recognized as an important research problem to study as a topic independent of application area (Madhavan et al., 2001). The main purpose of this phase is to recognize not only similarities but also differences between two source schemata. In our approach we have decided to adopt what Rahm and Bernstein (2001) called a composite schema based matching approach. It is schema based since we are working with conceptual schemata and composite since we are combining the results from several matchers into one matching result. This is motivated since using several matching approaches and combining them into one result should produce a better matching result than the using of one single matching approach (Rahm and Bernstein, 2001). Before presenting our approach to the comparison of the schemata, a comment of the usage of schema and instance based matching is needed. Using EM it is possible, through the classification dependency, graphically to illustrate that one concept is an instantiation of another concept (see Figure 1). This does not, however, mean that the concept that is an instance-of another concept is what Rahm and Bernstein (2001) call *instance-level data*. We therefore classify our approach as a composite schema based matching approach. Finally, in our approach the comparison of schemata is divided into two tasks: comparison of the concepts as such (also known as element level) and comparison of the concepts neighborhood (also known as structural level).

In our approach we begin with *concept name comparison* in which the names of the concepts in one schema are compared with the names of the concepts in another schema. Such a comparison might yield the following results: **Match,** the concept names are the same, **No Match,** the concept names are not the same, and **Partly Match,** the whole or parts of one concept name is part of another concept name. If comparison of concept names results in a **Partly Match** we continue and apply two so called linguistic rules (Bellström and Vöhringer, 2011a) in which it is indicated if one concept '**belongs-to**' another concept, if one concept is '**related-to**' another concept or if one concept '**is-a**', a specialization, of another concept. In Bellström and Vöhringer (2011a) we formulated these two rules as follows:

> *A.* If the compared concept names are available in the form of A and AB (i.e., A corresponds to the compound AB minus the head B), then the relationship "AB belongs/related to A" can be assumed.
>
> *B.* If the compared concept names are available in the form B and AB, where A is the head of the compound AB, then the relationship "AB is a B" can be assumed. (p. 26)

If comparison of concept names results in **No Match,** we continue and check concept definitions if these are available. Checking the concept definitions might result in **No Match** or **Synonyms**. Finally, as addressed in Bellström and Vöhringer (2011b), if domain ontology exists, it is possible to carry out a domain ontology-based comparison of concept names.

After having focused on the concepts as such, we continue and perform comparison of concepts neighborhood. In our approach concept neighborhood is defined as the directly connected concepts to the concept that is currently analyzed for similarities and differences. However, before conducting comparison of concepts neighborhood several influence factors such as *polysemy count* (the number of meanings a concept has in a given language), *valency* (the number of parameters, other words, a concept needs to get its meaning in a given language) and *domain weight* (each concept might have a specific weight, a number that has been given manually by the domain experts) should be taken into account since these might indicate if comparison of concepts neighborhood is even necessary to perform (Bellström and Vöhringer, 2009).

If it is decided to go ahead with comparison of concepts neighborhood, which is often the case since we only get indications from the influence factors, we first apply a set of rules. The usage of the rules might yield the following results: **Match,** the concepts neighborhood are the same, **No Match,** the concepts neighborhood are not the same

and finally, **Partly Match,** parts of the concepts neighborhood are the same but some still differ.

The rules were first presented in Bellström (2010a) and later on adapted and modified in Bellström (2010b) and Bellström and Vöhringer (2011a; 2011b). In total, we apply seven rules when we have a **Match** of concept names and four rules when we have a **Partly Match** of concept names. The seven rules we apply for comparison of concepts neighborhood when we have a **Match** of concept names are as follows. If comparison of concept names yields **Match** and comparison of concepts neighborhood yields:

(1) **Match** THEN *equivalent* concepts can be assumed.

(2) **No Match** THEN *homonymic* concepts can be assumed.

(3) **Partly Match** THEN *synonymic* concepts can be assumed IF one concept in each source schemata is named differently

(4) **Partly Match** THEN an *association dependency* can be assumed IF one concept is named with a following addition to the other one AND cardinality between these concepts is one-to-one.

(5) **Partly Match** THEN a *hypernym-hyponym dependency* can be assumed IF one concept is named with prior addition to the other one.

(6) **Partly Match** THEN *a holonym-meronym dependency (composition)* can be assumed IF one concept is named with a following addition to the other one AND cardinality between these concepts is one-to-many.

(7) **Partly Match** THEN *a holonym-meronym dependency (aggregation)* can be assumed IF one concept is named with a following addition to the other one AND cardinality between these concepts is zero-to-many.

The four rules we apply for comparison of concepts neighborhood when we have a **Partly Match** of concept names are as follows:

(1) IF comparison of concept names yields **Partly Match**, one concept is named with a following addition to the other one, AND comparison of concepts neighborhood yields **Partly Match** or **Match** THEN *an association dependency* can be assumed IF cardinality between these concepts is one-to-one.

(2) IF comparison of concept names yields **Partly Match**, one concept is named with a follow addition to the other one, AND comparison of concepts neighborhood yields **Partly Match** or **Match** THEN *a holonym-meronym dependency (composition)* can be assumed IF cardinality between these concepts is one-to-many.

(3) IF comparison of concept names yields **Partly Match**, one concept is named with a follow addition to the other one, AND comparison of concepts neighborhood yields **Partly Match** or **Match** THEN *a holonym-meronym dependency (aggregation)* can be assumed IF cardinality between these concepts is zero-to-many.

(4) IF comparison of concept names yields **Partly Match**, one concept is named with a prior addition to the other one, AND comparison of concepts neighborhood yields **Partly Match** or **Match** THEN a *hypernym-hyponym dependency* can be assumed.

In Bellström (2012) we also proposed two rules for the recognition of power types and homonyms that are applicable when integrating schemata that are designed on different levels of abstraction (schema level and instance level) and includes the classification dependency (see Figure 1) in at least one of the input schemata. A power type is by OMG (OMG UML, 2011) described as “a class whose instances are subclasses of another class” (p. 76) indicating that a power type includes two levels of abstraction. The rules for recognition of power types (1) and homonyms (2) might shortly be summarized and explained as follows:

(1) If comparison of concept names yields **Match** between three concepts, e.g. S1.A = S2.A, S1.AB = S2.AB and

https://openaccess.cms-conferences.org/#/publications/book/978-1-4951-2091-6

S1.C = S2.C, and comparison of concepts neighborhood yields **Partly Match** meaning, cardinality is the same between the concepts A and AB in S1 and S2, e.g. A many-to-one AB (one A might have one and only one AB while AB might have zero or many A), **AND** S1.C 'is-a' S1.A **AND** S2.C is 'instance-of' S2.AB **THEN** it can be assumed that AB is a *power type*. The rule is even stronger if AB is a composite name of concept name A with a following addition B.

(2) If comparison of concept names yields **Match** between three concepts, e.g. S1.A = S2.A, S1.AB = S2.AB and S1.C = S2.C, and comparison of concepts neighborhood yields **Partly Match** meaning, cardinality is the same between the concepts A and AB in S1 and S2, e.g. A many-to-one AB (one A might have one and only one AB while AB might have zero or many A), **AND** S1.C 'is-a' S1.A **AND** S1.C is 'instance-of' S1.AB **AND** S2.C is 'instance-of' S2.A THEN it can be assumed that C is a *homonym*. Also this rule might be stronger if AB is a composite name of concept name A with a following addition B.

In Batini et al. (1986) the authors divide conflicts into name conflicts (homonyms and synonyms) and structural conflicts (type, dependency, key and behavioral conflicts). In our approach a *dependency conflict* might also appear. A dependency conflict can be assumed if comparison of concept names yields **Match** between two concepts and comparison of concepts neighborhood yields **No Match** between the same concepts.

Finally, as discussed in Bellström and Vöhringer (2011a; 2011b), if domain ontology exists it might be used for neighborhood comparison. Besides this, taxonomy based matching using the Lesk algorithm (Lesk, 1987; Banerjee and Pederson, 2002) might also be used as a matching approach during neighborhood comparison (Vöhringer and Fliedl, 2011).

Since we are applying what Rahm and Bernstein (2001) called a composite schema based matching approach, we also take into account the result of all the used matching approaches. In doing so, it is decided if and how the input schemata concepts are similar or different. The recognized and documented similarities and differences are now passed on to the following phase in the schema integration process in which they are resolved.

Conforming the Schemata

Conforming the schemata is the third phase of the integration process and this phase has also received a lot of attention by the research community. For instance, Spaccapietra and Parent (1994) mention this phase as a key issue and Lee and Ling (2003) as the most critical issue of schema integration. Nevertheless, depending on the chosen modeling language and the modeled level of abstraction the right resolution methods should be chosen not only to improve the schema quality (Bellström and Kop, In press) but also to prevent that semantic loss occurs (Bellström, 2009). Since the EM schemata are implementation-independent, several resolution methods for the recognized similarities and differences differ compared with schemata that are implementation-dependent such as the ER and the UML.

In Bellström (2010a) a summary of resolution methods of so-called linguistic conflicts for EM are presented and illustrated. More specifically this means that for *homonyms* prefixing together with the inheritance dependency is the chosen solution. For *synonyms* the mutual inheritance dependency is the chosen solution and for *cyclic generalization* (Song, 1995) and *reverse subset relationships* (Batini et al., 1992) more precise concept names (including the original ones) should be introduced. For the *inter-schema hypernym-hyponym property* the inheritance dependency should be used when the concepts are not exclusive, and the subset-of dependency when the concepts are exclusive. Finally, for the *inter-schema holonym-meronym property* the aggregation dependency should be used when the dependency includes a zero-to-many cardinality and the composition dependency when the dependency includes a one-to-many cardinality. However, it should be noted that the recognized inter-schema hypernym-hyponym properties and the inter-schema holonym-meronym properties should be documented and passed to the last phase of the integration process in which it should be used as a guidance (knowledge repository), while merging the input schemata and restructuring the integrated schema (Bellström and Kop, In press).

If a dependency conflict has been recognized, a one is chosen over zero and many over one on the cardinality. This is motivated since changing a one into a zero, thus introducing a weaker dependency, might instead change the meaning of the dependency for one or several domain experts.

In Bellström (2012) the problem of integrating schemata on two different levels of abstraction (schema level and instance level) was addressed. In this context this involves how to resolve recognized power types and homonyms that include not only the classification but also inheritance dependency. Resolving *power types* both the instance-of and the inheritance dependency are included and resolving *homonyms*, more precisely concept names (including the original ones), are introduced together with a new instance-of dependency.

Inference rules (Gustas, 2005) might also be used while resolving several of the recognized similarities and differences (Bellström, 2006b; Bellström, 2012). Applying inference rules makes it possible not only to deduce new dependencies but also new concepts from already existing ones.

Finally, independent of resolution method, it is important that the recognized similarities and differences are resolved without losing any of the concepts (including their names) and/or dependencies since these might actually mean something to one or several of the domain experts. Ignoring this could instead cause semantic loss (Bellström, 2009), which in the end could cause the integrated schema to be incomplete and hard to understand for the domain experts.

Merging and Restructuring

Merging and restructuring is the fourth and last phase of the integration process. In this phase the input schemata are first merged, resulting in a first intermediate schema, and later on restructured to fulfill several quality criteria (Batini et al., 1986; Batini et al., 1992) and/or quality factors (Moody and Shanks, 2003). Schema merging has also been recognized as an important research problem to study as a more generic topic (Pottinger and Bernstein, 2003; Quix et al., 2007). To aid in merging and restructuring the earlier recognized inter-schema properties should also be used as guidance; they can even be viewed as a knowledge repository (Bellström and Kop, In press). In EM we merge the two source schemata and restructure the integrated schema based on the following reasoning.

If two concepts are recognized as *independent*, they are both included in the integrated schema.

If two concepts are recognized as *equivalent*, they are merged into one concept and included in the schema.

If two concepts are recognized as *synonyms*, they are both included in the integrated schema with a mutual inheritance dependency placed in between.

If two concepts are recognized as *homonyms*, they are both included in the integrated schema. However, the concept names should have been resolved by either giving one or both concepts more precise names and/or giving the concept names a prefix and introducing an inheritance dependency between the prefixed concept (the specialization) and the original concept (the generalization).

If two concepts are recognized as having a *hypernym-hyponym dependency* (is-a), both concepts are included in the integrated schema and either an inheritance dependency or a subset-of dependency is drawn between them. The documented inter-schema properties should be used to decide which dependency to use. The concepts that are truly redundant (the hypernym and hyponym might have concepts in common) are thereafter deleted since they might instead be inherited.

If two concepts are recognized as having a *holonym-meronym dependency* (part-of), both concepts are included in the integrated schema and either a composition dependency or an aggregation dependency is drawn between them. The documented inter-schema properties should be used to decide which dependency to use.

Finally, it should be noted that the last phase, merging and restructuring, may well partly overlap with the former phase, conforming the schemata, not only since both phases are dealing with issues on how to actually merge the schemata, but also since both phases are dealing with issues on how to counter the occurrence of semantic loss.

A SMALL AND ILLUSTRATIVE EXAMPLE

In this section we give a very small and illustrative example of how parts of the described and discussed semi-

automatic approach for the integration of structural EM schemata could be applied. More specifically, this means how to semi-automatically integrate schema one (S1) and schema two (S2) in Figure 3a.

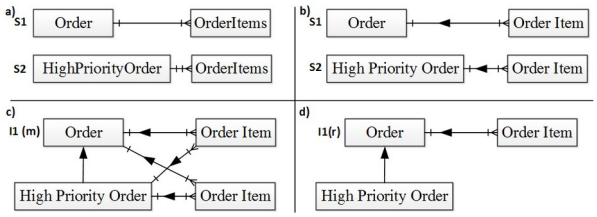

Figure 3. A small and illustrating example

Our example starts with pre-integration and the task of adapting the concept names used in the input schemata (S1 and S2). In doing so, 'OrderItems' in S1 is altered to 'Order Item'. In S2 'HighPriorityOrder' is altered to 'High Priority Order' and 'OrderItems' to 'Order Item'. In other words, compounded concept names are expanded and concept names are reduced to their base form. However, the concept names used in Figure 3a are of course saved not only to prevent semantic loss but also for reasons of traceability since the process should be reversible. Moreover, in this specific case a stemmer is not used. The last task for the pre-integration phase that we conduct in our example is to analyze the input schemata for intra-schema conflicts. In doing so, a minor version of phase two and three of the integration process is conducted. In both S1 and S2 we have a Partly Match of concept names and therefore continue to apply the first linguistic rule. For S1 this means that 'Order Item' belongs/is related to 'Order' and for S2 this means that 'Order Item' belongs/is related to 'High Priority Order' which is also illustrated in Figure 3a. However, continuing with the comparison of concepts neighborhood and the second rule for Partly Match of concept names, it can be assumed, since there is one-to-many cardinality, that a holonym-meronym dependency (composition) exists between 'Order' and 'Order Item" and, between 'High Priority Order' and 'Order Item', resulting in the new versions of the schemata illustrated in Figure 3b.

We now continue with the comparison of the schemata by comparing the concept names of each schema. The comparison of concepts names results in one Match, 'S1.Order Item' = 'S2.Order Item', and three Partly Match 'S1.Order' ~ 'S2.Order Item', 'S2.High Priority Order' ~ 'S1.Order Item' and 'S1.Order' ~ 'S2.High Priority Order'. Since we have Partly Match of concept names, we continue and apply the two linguistic rules resulting in the following: 'S2.Order Item' belongs/related to 'S1.Order', 'S1.Order Item' belongs/related to 'S2.High Priority Order' and 'S2.High Priority Order' is-a 'S1.Order'. After comparing the concept names as such we continue with the comparison of concepts neighborhood.

Since the comparison of concept names resulted in one Match 'S1.Order Item' = 'S2.Order Item' we continue and check concepts neighborhood with the seven rules for Match of concept names. In doing so, rule three and five are applicable. Rule three states that it can be assumed that 'S1.Order' and 'S2.High Priority Order' are synonymous concepts while rule five states that it can be assumed that 'S2.High Priority Order' is-a 'S1.Order' (a hypernym-hyponym dependency). Comparison of concepts names also resulted in three Partly Match and we therefore also continue and check concepts neighborhood with the four rules for Partly Match of concept names. In doing so, rule two is applicable for 'S1.Order' ~ 'S2.Order Item' and 'S2.High Priority Order' ~ 'S1.Order Item', which states that a holonym-meronym dependency (composition since the cardinality is one-to-many) can be assumed between the two concepts. Rule four is applicable to 'S1.Order' ~ 'S2.High Priority Order', which states that a hypernym-hyponym dependency (is-a) can be assumed between the two concepts.

In conforming the schemata, the recognized conflicts are resolved and the recognized inter-schema properties are documented and passed to the last phase in which they are applied as a guidance while merging and restructuring the integrated schema. In our example we did not recognize any conflicts. Although a synonym conflict was indicated, we decided that 'S2.High Priority Order' is-a 'S1.Order' since several of the used comparison methods indicated this. Nevertheless, three inter-schema properties were recognized: 'S1.Order Item' part-of (composition) 'S2.High Priority Order', 'S2.Order Item' part-of (composition) 'S1.Order' and 'S2.High Priority Order' is-a 'S1.Order' which are also passed to the last phase of the integration process.

In merging and restructuring, we first merge S1 and S2 in Figure 3b resulting in the integrated intermediate schema

in Figure 3c. However, the integrated schema contains redundant concepts and dependencies that we need to get rid of. Since 'High Priority Order' is-a 'Order' and therefore also inherit 'Order Item', we can get rid of the redundant 'Order Item' concept that was part of S2 from the beginning. In doing so, we also get rid of three composition dependencies (part-of), resulting in the schema in Figure 3d, which is also in this case the end result of the integration process. Finally, even though the example is very small it still indicates how complex schema integration might be and how the described and discussed semi-automatic approach might facilitate the integration process.

SUMMARY AND FUTURE WORK

In this paper we have described and discussed a semi-automatic approach for the integration of structural Karlstad EM Schemata. In doing so, not only the automatic but also the manual tasks that are applicable in the integration process have been addressed. The research approach has been inspired by design science and the main contribution can be classified as a method.

In future work we plan to address not only mandatory and optional tasks but also the order of tasks to be performed in the schema integration process. We also plan to develop a prototype in which performance of the prototype is closely related to both the order of tasks and mandatory and optional tasks. The human computer interaction (HCI) aspects of the graphical user interface (GUI) that the domain expert interacts with in the prototype are also important to consider for future work. While addressing the HCI aspects of the prototype, we plan to use Ozlab (Pettersson and Wik, 2013), a tool that makes it possible to test the GUI before any implementation has been conducted.

REFERENCES

Banerjee, S. & Pederson, T. (2002). An adapted Lesk algorithm for word sense disambiguation using WordNet. in: A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing* (pp. 136-145).

Batini, C. & Lenzerini, M. (1984). A methodology for data schema integration in the Entity-Relationship model. *IEEE Transactions on Software Engineering*, *10*(6), 650-664.

Batini, C., Ceri, S. & Navathe, S.B. (1992). *Conceptual database design An Entity-Relationship approach*. Redwood City: The Benjamin/Cummings Publishing Company.

Batini, C., Lenzerini, M. & Navathe, S.B. (1986). A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, *18*(4), 323-364.

Bellström, P. (2006a). *View integration in conceptual database design: Problems approaches and solutions* (Karlstad University Studies, No. 2006:5). Licentiate Thesis, Karlstad: Karlstad University.

Bellström, P. (2006b). Bridging the gap between comparison and conforming the views in view integration. in: Y. Manolopoulos, J. Pokorný & T. Sellis (Eds.), *Local Proceedings of the 10th East-European Conference on Advances in Databases and Information systems* (pp. 184-199).

Bellström, P. (2009). On the problem of semantic loss in view integration. in: C. Barry, K. Conboy, K. Lang, G. Wojtkowski, & W. Wojtkowski (Eds.), *Information Systems Development Challenges in Practice, Theory, and Education Volume 2* (pp. 963-974).

Bellström, P. (2010a). *Schema integration How to integrate static and dynamic database schemata* (Karlstad University Studies, No. 2010:13). Dissertation, Karlstad: Karlstad University.

Bellström, P. (2010b). A rule-based approach for the recognition of similarities and differences in the integration of structural Karlstad Enterprise Modeling schemata. in: P. van Bommel, S. Hoppenbrouwers, S. Overbeek, E. Proper & J. Barjis (Eds.), *The Practice of Enterprise Modeling* (pp. 177-189).

Bellström, P. (2012). Aspects of the classification dependency in the integration of structural Karlstad Enterprise Modeling schemata. in: *Proceedings of the European Conference on Information Systems ECIS 2012, Paper 32*.

Bellström, P. & Kop, C. (2012). Towards a framework for schema quality in the schema integration process. in: K. Sandkuhl, U. Seigerroth & J. Stirna (Eds.), *Short Paper Proceedings of the 5th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling*.

Bellström, P. & Kop, C. (In press). Towards quality driven schema integration process tasks. *The Sixth International Conference on Information, Processing, and Knowledge Management*.

Bellström, P., Kop, C. & Vöhringer, J. (2012). Semi-automatic schema pre-integration in the integration of modeling language independent behavioral schemata. in: J. Lloret Mauri & P. Lorenz (Eds.), *The Forth International Conference on Information, Processing, and Knowledge Management* (pp. 12-17).

Bellström, P., Vöhringer, J. & Kop, C. (2008). Guidelines for modeling language independent integration of dynamic schemata. in: C. Pahl, (Ed.), *Proceedings of the IASTED International Conference on Software Engineering* (pp. 112-117).

Bellström, P. & Vöhringer, J. (2009). Towards the automation of modeling language independent schema integration. in: A. Kusiak & S. Lee (Eds.), *International Conference on Information, Process, and Knowledge Management* (pp. 110-115).

Bellström, P. & Vöhringer, J. (2011a). A three-tier matching strategy for predesign schema elements. in: B. Jerman-Blazic & D. Dan Burdescu (Eds.), *The Third International Conference on Information, Process, and Knowledge Management* (pp. 24-29).

Bellström, P. & Vöhringer, J. (2011b). A semi-automatic method for matching schema elements in the integration of structural pre-design schemata. *International Journal on Advances in Intelligent Systems*, *4*(3 & 4), 410-422.

Bernstein, P.A., Madhavan, J. & Rahm, E. (2011). Generic schema matching, ten years later. in: *Proceedings of the VLDB Endowment* (pp. 695-701).

Convent, B. (1986). Unsolvable problems related to the view integration approach. in: *Proceedings of Database Theory* (pp. 141-156).

Doan, A., Noy, F.N. & Halevy, A.Y. (2004). Introduction to the special issue on semantic integration. *SIGMOD Record, 33*(4), 11-13.

Ekenberg, L. & Johannesson, P. (1996). A formal basis for dynamic schema integration. in: B. Thalheim (Ed.), *Conceptual Modeling* (pp. 211-226).

Gruber, T.R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, *5*, 199-220.

Gustas, R. (2005). Inference rules of semantic dependencies in the Enterprise Modelling. in: H. Fujita & M. Mejri (Eds.), *New Methods in Software Methodologies, Tools and Techniques* (pp. 235-251).

Gustas, R. & Gustiené. P. (2004). Towards the enterprise engineering approach for information system modelling across organisational and technical boundaries. in: O. Camp, J. Filipe, S. Hammoudi & M. Piattini (Eds.), *Enterprise Information Systems V* (pp. 204-215).

Gustas, R. & Gustiené. P. (2008). Pragmatic-driven approach for service-oriented analysis and design. in: P. Johannesson & E. Söderström (Eds.), *Information Systems Engineering: From Data Analysis to Process Networks* (pp. 97-128). London: IGI Global.

Hevner, A. (2007). A three cycle view on design science research. *Scandinavian Journal of Information Systems*, *19*(2), 87-92.

Hevner, A. & Chatterjee, S. (2010). *Design research in information systems theory and practice*. New York: Springer.

Hevner, A., March, S.T., Park, J. & Ram, S. (2004). Design science in information systems research. *MIS Quartely 28*(1) 75-105.

Hull, D.A. (1996). Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society for the Information Science*, *47*(1), 70-84.

Iivari, J. (2007). A paradigmatic analysis of information systems as a design science. *Scandinavian Journal of Information Systems*, *19*(2), 39-64.

Johannesson, P. (1993). *Schema integration, schema translation, and interoperability in federated information systems*. Dissertation, Stockholm: Stockholm University.

Lee, M.L. & Ling, T.W. (2003). A methodology for structural conflict resolution in the integration of Entity-Relationship schemas. *Knowledge and Information System*, *5*(2) 225-247.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. in: *Proceedings of the 5th annual international conference on Systems documentation*.

Madhavan, J., Bernstein, P.A. & Rahm, E. (2001). Generic schema matching with Cupid. in: *Proceedings of the 27th VLDB Conference* (pp. 49-58).

March, S.T. & Smith, G.F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*, 251-266.

Miller, G.A. (1995). WordNet: A lexical database for English. *Communication of the ACM*, *38*(11), 39-41.

Moody, D.L. & Shanks, G.G. (2003). Improving the quality of data models: Empirical validation of a quality management framework. *Information Systems Journal*, *28*(2), 619-650.

Navathe, S., Elmasri, R. & Larson, J. (1986). Integrating user views in database design. *IEEE Computer*, *19*(1), 50-62.

OMG UML. (2011). Object Management Group: OMG Unified Modeling Language (OMG UML) Superstructure. Retrieved from http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF/

Persson, A. & Stirna, J. (2010). Towards defining a competence profile for the enterprise modeling practitioner. in: P. van Bommel, S. Hoppenbrouwers, S. Overbeek, E. Proper & J. Barjis (Eds.), *The Practice of Enterprise Modeling* (pp. 232-245).

Pettersson, J. S. & Wik, M. (2013). Designing through testing: A tool for finding usable mobile services. Retrieved from http://www.kau.se/sites/default/files/Dokument/subpage/2010/08/designing_through_testing_ozlab_info_nov_2013_pd_71499.pdf.

Pottinger, R.A. & Bernstein, P.A. (2003). Merging models based on given correspondences. in: *Proceedings of the 29th VLDB conference* (pp. 862-873).

Quix, C., Kensche, D. & Li, X. (2007). Generic schema matching. in: J. Krogstie, A. Opdahl, G. Sindre (Eds.), *Advanced Information Systems Engineering*, (pp. 127-141).

Rahm, E. & Bernstein, P.A. (2001). A Survey of approaches to automatic schema matching. *The VLDB Journal, 10*, 334-350.

Song, W. (1995). *Schema integration – Principles, methods, and applications*. Dissertation, Stockholm: Stockholm: University.

Song, W. (1997). Semantic knowledge acquisition and computation in conceptual schema integration. in: H. Kangassalo, J.F. Nilsson, H. Jaakola & S. Ohsuga (Eds.), *Information Modelling and Knowledge Bases VIII* (pp. 199-212).

Sorrentino, S., Bergamaschi, S., Gawinecki. M. & Po, L. (2009). Schema normalization for improving schema matching. in: A. H. F. Laender, S. Castano, U. Dayal, F.Casati, F. & J.P.M. de Oliveira, (Eds.), *Conceptual Modeling* (pp. 280-293).

Spaccapietra, S. & Parent, C. (1994). View integration: A step forward in solving structural conflicts. *IEEE Transactions on Knowledge and Data Engineering*, *6*(2), 258-274.

Stumptner, M., Schrefl, M. & Grossmann, G. (2004). On the road to behavior-based integration. in: *Proceedings of the first Asian-Pacific conference on Conceptual modelling Volume 31* (pp. 15-22).

Vöhringer, J. (2010). *Schema integration on the predesign level*. Dissertation, Klagenfurt: Alpen-Adria-Universität Klagenfurt.

Vöhringer, J. & Fliedl, G. (2011). Adapting the Lesk algorithm for calculating term similarity in the context of requirements engineering. in: J. Pokorny, V. Repa, K. Richta, W. Wojtkowski, H. Linger, C. Barry, & M. Lang (Eds.), *Information Systems Development Business Systems and Services: Modeling and Development* (pp. 781-790).