Schema Quality Improving Tasks in the Schema Integration Process

Peter Bellström
Information Systems
Karlstad University
Karlstad, Sweden
e-mail: peter.bellstrom@kau.se

Christian Kop
Institute for Applied Informatics
Alpen-Adria-Universität Klagenfurt
Klagenfurt, Austria
e-mail: chris@ifit.uni-klu.ac.at

***Abstract* — In this article, we address quality in the schema integration process. More specifically, we focus on schema quality improving tasks in the schema integration process. In doing so we describe best practices found in literature used for conceptual modeling as such and apply these to schema integration tasks. Particularly, we address five tasks within the integration process that if used with best quality practices should improve the quality of the integrated schema. Within each best practice, we also address the usage of knowledge repositories to aid in the process of creating a high quality integrated schema. The five tasks are as follows: choosing the right integration strategy, choosing the right conflict resolution methods for the chosen level of abstraction, introducing inter-schema properties to improve and clarify dependencies, combining methods, approaches and guidelines to facilitate recognition of conflicts and finally restructuring. The main contribution is given in the terms of which best practices in conceptual modeling are combined with important integration tasks.**

Keywords - Information and Model Management, Organizational Information, Schema Integration, Schema Integration Process, Schema Quality, Best Practice for Conceptual Modeling

I. INTRODUCTION

Schema integration has a long research tradition. Nevertheless, it is still relevant and many tasks of the schema integration process are needed all the time. There are several reasons and application scenarios, which emphasize this prediction.

Schemata are not built from scratch nowadays. There are many schemata available on the Web. Several of these schemata can be (re)used. Reusing one means that it must be aligned and if needed finally integrated with existing schemata.

Enterprises involve a great deal of data, which constitute an important economic resource and have to be maintained carefully. From an economic point of view, data can be classified into master data, inventory data and transaction data. Especially master data can be used in different information systems within an enterprise and thus shared across this organisation. Integration of data schemata (e.g., schemata of master data) becomes necessary if, for instance, two enterprises merge. A data schema that is used in several enterprises has to be at least aligned too. For another application scenario where integration can take place, we can consider an international operative enterprise with branches in different countries. It has to be expected that the branches will show a tendency to generate proprietary schema parts, important only for this branch. Therefore, the schema will evolve over time. Since the master data schema has to be shared by the whole enterprise, it is necessary to integrate new information consistently into the existing enterprise master data schema.

Finally, if enterprises use provided Web Services, then it might be good to know the business process model and at least match the business process models and data models and check for compliance of the models to the Web Service with the respective models of the enterprise.

A good quality of the result in such contexts is very important. Literature on quality mainly focuses on the quality of the product (i.e., the model). The criteria a model must fulfill in order to have a certain quality is explained. In order to achieve this quality one must also look at the process and think about improving the process tasks.

In [1], we gave a first description of what can be done in the integration process of static schemata in order to get a good integrated model (schema). In this article, we present and describe a continuation of this work. Since an integrated schema is a schema too, we analyzed the literature focusing on static modeling and the kind of process tasks that lead to a better quality of the schema. Then we applied strategies to tasks that have to be done in the integration process. Particularly, we address five tasks within the integration process that if used with best quality practices should improve the quality of the integrated schema. The five tasks are as follows: choosing the right integration strategy, choosing the right conflict resolution methods for the chosen level of abstraction, introducing inter-schema properties to improve and clarify dependencies, combining methods, approaches and guidelines to facilitate recognition of conflicts and finally restructuring.

Since the paper covers the schema integration process and the quality of the integrated schema and the process, this paper is structured as follows. In Section II, we give an overview on integration approaches and quality of schemata. In Section III, we describe the integration process. Section IV focuses on some best practices for improving schema quality. In Section V, we describe the impact of the best practices mentioned in Section IV on the five tasks mentioned in Section III. The paper closes with conclusion and future work.

II. Related Work

In this section, we first address related work in relation to schema integration. This is followed by related work in relation to schema quality. Finally, the section ends with a short summary and a discussion on the research gap that we are addressing.

A. Integration

There is a long research history on several aspects of schema integration. A first substantial work, on integration was made by [2] in the mid-1980s. In another work by [3], other approaches on integration were summarized. In the following years other integration approaches have been published, which focus on several aspects of the integration problem.

In [4], the authors used attribute equivalence as the most basic concept to explain the integration of structural schemata. In [5], the authors presented operators for deciding on the similarity or dissimilarity of schema construct. On the basis of defined assertions, in [6], the author proposed a method to detect equivalent schemata and to automatically integrate two schemata. In [7], the authors concentrate on the automatic detection of naming conflicts. Further algorithms for structural schema integration can be found in [8]. In [9], the authors integrate semantically enriched database schemata. In [10], the author presented an object-oriented framework for the integration of heterogeneous databases. In [11], the authors introduced linguistic knowledge for the integration step. For relationships, for instance, verbs can name relationships. The knowledge about the verbs and their linguistic semantic roles support the integration. In [12], the authors described black board architecture for schema integration of existing databases. With the system, knowledge from designers and end users, which feed the system is shared. The impact of similarity measures for schema matching and data integration is discussed in [13]. In [14], the authors described the integration of state charts in object-oriented models. The work of [15] is based on the formalization of state chart constructs. In [16], the authors proposed a meta-class framework, on which integration should be based. In [17], the author gave an overview of business process integration. In [18], the authors proposed OWL-S ontologies as a support for business process integration. In [19], the authors described the integration of use cases on the basis of petri net models. Finally, in [20] the authors used a behavior tree approach for integrating requirements.

B. Schema Quality

A great deal of work has been done on the quality of conceptual schemata (models) too. Although quality is a feature of a product or artifact (e.g., a schema), it is also necessary to think about the quality of the process of generating the product to support the quality of the product.

In [21], the authors have listed eight schema quality characteristics: completeness, correctness, expressiveness, readability, minimality, self-explanation, extensibility, and normality. In [22], a framework consisting of three dimensions is proposed: “syntax”, “semantic” and “pragmatics”. The syntax-dimension reflects the vocabulary and grammar (i.e., meta-model) of a schema. The semantic dimension relates the used terms and notions to the domain context. The chosen notions modeled by modeling elements must be legitimate and relevant in the domain, and they must be relevant and legitimate to the purpose for which the schema has been built. Finally, the pragmatic dimension is achieved if the audience can understand and follow the schema.

In [23], the author concluded that there is still a need for standards, which are also accepted by the industry.

In [24], the authors focused on process quality for the development of data schemata (ER diagrams). Their approach was evaluated in a large Australian bank. In the empirical study, it was also important, that the quality was checked throughout the schema development process. In particular, quality-checking was not only made at the end of a phase but before, during and after the schema development phases. Furthermore, it turned out, that an information architect, who checks the schema with respect to enterprise terms, can support the quality.

In [25], the authors presented a framework of four quality characteristics for an ER modeling language: clarity, simplicity, expressiveness, and minimality.

In [26], the authors described the “Guidelines of Modeling (GoM)”. Six principles of modeling are introduced in this framework: correctness, relevance, economic efficiency, clarity, comparability and systematic design. These principles can be seen as general strategic and objective definitions for modeling. Based on these goals, the concluded modeling process consisted of the following steps: goal definition, construction of an overall navigation and structural framework, modeling as such, and completion and consolidation.

With the **se**miotic **qual**ity framework (SEQUAL), [27] explains quality of models with model externalization, goals of modeling, modeling domain, explicit knowledge of social actors, interpretation of the social actors and technical actors as well as with language extension.

C. Summary of the Literature

We adopted the integration process as described in [3], since this is a well-established process. They divided the integration process into four phases: pre-integration, comparison of the schemata, conforming the schemata and merging and restructuring. In Section III, we describe this process in more detail.

In Section IV, we continue the description about schema quality according to some selected best practices out of the list of schema quality approaches. We have chosen these approaches since they have shown in practice that they improve schema quality. In Section V, we will then take specific best practices and combine them into five tasks of the integration process steps described in Section III.

III. Integration Process

This section should be viewed as a reference point for the following sections, in which we describe and discuss best practices in the schema integration process. The integration process starts with a set of schemata, often referred to as views. These views are integrated in order to evolve the global schema. The schema evolvement takes place in the four phases proposed in [3], starting with pre-integration (A), followed by comparison of the schemata (B), and conforming the schemata (C), ending with merging and restructuring (D). The output of one phase is used as the input of the next phase (see Figure 1).

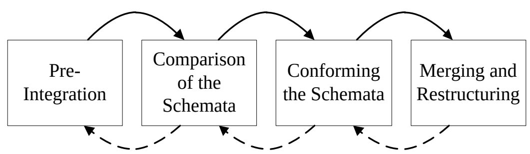

Figure 1. The Schema Integration Process (adapted and modified from [28] p. 20)

A. Pre-Integration

Several tasks should be carried out in the first phase of schema integration. In [29], the author mentions that translating all schemata into the chosen modeling language, checking for differences and similarities in each schema and selecting the integration strategy are all tasks to be performed in pre-integration. Three additional tasks to perform in pre-integration are proposed in [30] as follows: schema element name adoption, schema element disambiguation and introduction of missing relationships.

The output from this phase is a set of revised schemata, the definitions of schema elements and the chosen integration strategy.

B. Comparison of the Schemata

The second phase of schema integration has been researched a great deal and it has been called an important [29] and difficult phase [31][32]. Several authors [3][29][33] have assigned the following tasks to this phase: recognition of name conflicts, recognition of structural conflicts and recognition of inter-schema properties.

The output from this phase is a description of schema element similarities and a description of differences and a description of inter-schema properties.

C. Conforming the Schemata

Also conforming the schemata, phase three has received some attention by other researches. For instance, in [32], the authors called it the most critical phase and in [13] it was called the key issue in schema integration.

In conforming the schemata, the recognized similarities and differences are resolved by adjusting the input schemata.

The recognized inter-schema properties are also used in this phase. However, its full value is shown in merging and restructuring where they are used as a guideline while not only merging the schemata but also restructuring the integrated schema.

The output of this phase is a set of revised schemata.

D. Merging and Restructuring

The last task in the schema integration is merging and restructuring, in which the first task is to merge the revised input schemata into one global intermediate schema. The intermediate schema is then restructured, e.g., detected inter-schema properties are introduced to semantically enrich the schema. Furthermore, schema elements that are truly redundant are recognized and removed from the schema. Merging the schemata as well as restructuring the schema results in a new intermediate schema.

Before the integrated schema is handed over to the developers implementing the information system, the schema is again analyzed, meaning that the schema is checked and verified according to several quality criteria [3][21] and/or quality factors [24].

The result of this phase should be a high quality schema that can be passed to the following phases, in which the information system is implemented.

IV. Some Best Practices Regarding Schema Quality

Both the Guidelines of Modeling (GoM) [26] and the quality factors explained in [24] have a focus on improving the quality of the modeling process and quality of the resulting product (i.e., the conceptual schema).

Both frameworks are a good basis for understanding the quality of the conceptual modeling integration process. The Guidelines of Modeling provide a more strategic framework for covering all aspects of enterprise models (e.g., data, organization, processes, and behavior). The work in [24] focuses on data schemata (models), more specifically schemata modelled with ER diagrams.

Because of its operational focus, we adopted the following practices from [24] for the integration process in order to fulfill the quality factors and improve the quality of the modeling:

- Introducing a specific kind of stakeholder role – the information architect
- Introducing continuous quality checks and reviews.

As well as the general practices:

- Stakeholder participation
- Introducing naming conventions and standards

The information architect (in [24] called data administrator) is a person introduced to review a schema with respect to the other data schemata (models) existing in the enterprise.

According to the authors of [24], who proposed continuous checks and reviews for schema development, reviews must not only be made at the end but also before and during a development step. Such reviews should support the aim of the total quality management that the quality checks and reviews should NOT detect errors but prevent errors.

The participation of different kind of stakeholders is a successful technique used in Information Systems and Enterprise Engineering. Since the schemata (models) represent the knowledge of ideas of people with different backgrounds, it is necessary that different stakeholders are involved.

The introduction of an information architect implies also the usage and management of standards (e.g., what a schema should look like syntactically, what terms are used and preferred to other terms, etc.).

Beside this, the author of [27] also motivates the language as a factor for schema quality. According to [27] an important means to achieve good schema quality is to choose an appropriate language. He puts the language into the right perspective. A good language is useful but not sufficient. Someone can still generate a poor schema with a good language. Furthermore, the language is chosen already when modeling the original task. But a language has two aspects, one regarding its notions defined in the meta-model and the other aspect is the external representation of notions. In [49], the author proposed nine principles of language representation: Semantic clarity, Perceptual discriminability, Semantic Transparency, Complexity Management, Cognitive Integration, Visual Expressiveness, Dual Coding, Graphic Economy, Cognitive Fit. Semantic clarity means that there must be a one to one mapping between a representation and a notion. Perceptual discriminability is given if concepts are well distinguishable with their representation. A semantic transparency exists if the representation supports the meaning of the notion. With complexity management, the level of abstraction and filtering is supported by notion representations. A language has a cognitive integration if it is possible to navigate between subsets (i.e., different diagrams) of the language. Visual expressiveness describes to what extent the language cognitive variables (e.g., shape, size, color, brightness, etc.) support good interpretation and understanding of a schema. Does the language support graphical notation together with text (i.e., dual coding)? Are there not too many symbols for expressing notions of the language (graphic economy)? Finally, is it possible to adapt and use symbols, which were selected for the specific audience and the skills of the audience (i.e., cognitive fit)? Whereas the language itself cannot be changed, the external representation of the language can be changed in order to fit with the skills of the audience. The minimal change is the harmonization of external representations (semantic clarity), if there is no one-to-one mapping between the external representation and the notion and one notion has more representations. Another possibility would be to transform more abstract representations of notions to representations that fit with the skills of all users. Finally, representations can be changed in size and color to represent a certain state of a schema element (e.g., a concrete class). This would be a way to express that a certain schema element is already integrated.

The use of boundary objects can be another best practice especially for the communication between different kinds of stakeholders. In [34], the authors introduced boundary objects to communicate between professionals and amateurs in the zoological research field. Boundaries are abstracts that support the sharing of the knowledge and communication between communities of practice. Boundary objects are interfaces between these communities. In [35], the author described four classes of boundary objects: repositories, standardized forms and methods, physical objects like prototypes or models. Furthermore, he distinguishes between three types of knowledge boundaries between communities of practices: syntactic, semantic and pragmatic boundaries. Syntactic boundaries exist if different communities have different vocabularies. In this case, a common lexicon can support the overcoming of the differences. To solve semantic boundaries, the parties must create and define a common meaning with the help of boundary objects. Finally, boundary objects for pragmatic boundaries help to establish a negotiation process to find common interests. A pragmatic boundary always includes a semantic and a syntactic boundary. Also, a semantic boundary includes a syntactic boundary. In [36], enterprise architecture artifacts are introduced as boundary objects to support the communication and coordination in an enterprise transformation process.

Boundary objects should also be used during integration since many different kind of stakeholders are involved. Stakeholders can have different views on the domain and even different vocabularies and meanings. They also use parts of the schemata differently. Hence, schema integration has to solve even pragmatic boundaries. Therefore, boundary objects for pragmatic boundaries are needed.

Even the schema in the original modeling language can be a boundary object, because boundary objects can be schemata too. In the following, we will differentiate between a schema, the modeling language already in use for modeling and integration and boundary objects in a stricter sense. In this sense, a boundary object is any extension to the given schema or any additional method or schema from a different modeling language, if the original modeling language does not have sufficient power to act as an interlingua between all stake holders.

However, no predefined set of boundary objects exists, which fulfill the criteria to support communication. Therefore, the challenge is to find the adequate boundary object for a communication purpose. If there were several previous integration process projects in an enterprise, then the stakeholders can rely on given experiences. However, if it is the first integration project, then the stakeholders must agree on what kinds of models, objects, repositories or the like that they will use.

V. Applying Best Practices to Integration Tasks

In general, the best practice of “continuous improvement” is a driver for the whole integration process. Whereas quality is usually considered in or even after the last step of schema integration, we will follow the principle to introduce quality as early as possible. Therefore, we will focus on tasks needed during the whole integration process and not only in the last phases. We will relate the tasks to the best practices in order to improve them. These tasks are: choosing the right integration strategy, choosing the right conflict resolution methods for the chosen level of abstraction, introducing inter-schema properties to improve and clarify dependencies, combining methods, approaches and guidelines to facilitate recognition of conflicts and finally restructuring.

A. Choosing the Right Integration Strategy

In [3], several strategies are proposed to integrate end-user schemata (views). They distinguish between binary and n-ary integration strategies. Among the binary strategies, a ladder strategy or a balance strategy can be chosen. In the ladder strategy, the stakeholders start with two views. They integrate these two views. Then the first integrated schema is compared and matched with another view and so on. In the balanced strategy two views are integrated to become an intermediate schema. This intermediate schema is integrated with other intermediate schemata until the global schema is reached. The n-ary strategies are the one-shot strategy (a global schema is generated at once from all views) and the iterative strategy. The iterative strategy uses one shot strategies only to produce intermediate schemata. These schemata are then integrated with each other (two or more). Integrated schemata can also be integrated with further views. The iterative strategy can be seen as a mixture of the previous three strategies.

Figure 2 illustrates how the binary ladder strategy aids in the process of providing enough points of inspection during the schema integration process.

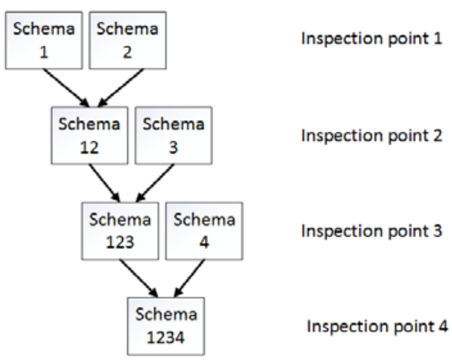

Figure 2. Binary ladder integration strategy with enough inspection points

1) Continuous Checks and Reviews

For continuous checks and reviews, the integration strategy must provide enough definite points of inspections.

A one shot strategy can be excluded as a good strategy by applying this best practice. Otherwise, it would mean that a global schema exists without any intermediate results. If intermediate results are missing, then it is impossible to define definite review milestones. Following the best practice of continuous improvement given in literature, an iterative and balanced or ladder strategy should be applied. Doing so, this intermediate schema can be reviewed each time an intermediate schema is generated.

It cannot be decided which of the other three strategies that should be chosen since all these strategies have intermediate points where schemata can be reviewed before or during integration. The decision between a balanced, a ladder, or an iterative strategy, is a pragmatic decision of available time for the integration and other environmental factors.

2) Information architect, stakeholder participation and standards

Since integration is part of modeling an information architect, standards and stakeholder involvement are also necessary for integration.

The information architect has to assure that a certain intermediate schema as well as the additional views already have to be integrated in compliance with existing schemata in the enterprise. Stakeholders check the semantic correctness and completeness with respect to a certain examined section represented by the views (schemata) or intermediate schemata. For both information architect and stakeholder involvement, strategies that have more intermediate points for discussions and reviews (i.e., ladder, balanced, iterative strategy) are more supportive.

Standards help to check if the schema is syntactically correct and if terms are used in compliance with the enterprise. It is therefore necessary that standards are used. Standards equally drive all the four strategies (one shot, ladder, balanced and iterative). Knowledge repositories, such as stemmers and lemmatizers, could be used to facilitate the task of checking that terms are used in a correct way. Drawing tools might also aid in the modelling process and be used to check that the schema is syntactically correct.

B. Choosing the Right Conflict Resolution Methods for the Chosen Level of Abstraction

In the second phase of schema integration, comparison of the schemata, two schemata are always compared to find the similarities and differences often referred to as conflicts. In the third phase that follows, conforming the schemata, the similarities and differences, i.e., the conflicts, are resolved. The problem is that the same resolution methods are often proposed (and used) for both implementation neutral schemata and implementation dependent schemata. However, using different conflict resolution methods for different levels of abstraction is very important since schemata are designed to be applied on different levels of abstraction. For instance, an implementation neutral schema is often used in the earlier phases of information systems development while an implementation dependent schema in the later phases is close to programming and technical issues.

The purpose of the schema under design may also vary. This is also addressed in [37], in which the authors state that “A schema can serve at least four different purposes. First, it can be used for clarifying the language used in an organisation. Secondly, it can be used for making explicit the rules that prevail in an organisation, which helps to criticise them and possibly to draw up new rules. Thirdly, a schema can be useful for reviewing existing information systems. Fourthly, a schema can be used for developing a new information system.” (p. 122).

One way to combine the two levels of abstraction, implementation neutral and implementation dependent, with the quotation given in[37] is described in Table I.

Summing up, if the differences between an implementation independent schema and an implementation dependent schema are ignored and the same conflict resolution methods are used, we might end up with not only a schema that is hard to understand but also end up with semantic loss. It is therefore of great importance that choosing the right conflict resolution methods for the chosen level of abstraction is taken into consideration in schema integration.

TABLE I. SCHEMA PURPOSE COMBINED WITH SCHEMA LEVELS OF ABSTRACTION

Purpose ([37])	Level(s) of abstraction	Comment
Clarifying the language used in an organisation.	Implementation neutral level	Often the designers are interested in concepts and connections between concepts and not in implementation dependent issues.
Making explicit the rules that prevail in an organisation.	Implementation neutral level	Rules must be expressed in a way that makes it possible for stakeholders to understand the rules and thus be able to criticize them.
Reviewing an already existing information system.	Implementation dependent level	In this case the schema should describe an already implemented information system.
Developing a new information system.	Implementation independent level / implementation dependent level	The designers might not only use different schemata during the development of the information system but also different modeling languages dependent on phase and focus in the information systems development process.

1) Continuous Checks and Reviews

Having compared two source schemata and recognized the conflicts between these, it is important that during the third phase, conforming the schemata, the right conflict resolution methods are used. Since this is not always the case, applying the best practice of continuous checks and reviews are of importance. If the designers and/or stakeholders introduce a resolution method that should not be used for the current level of abstraction, it should be recognized during continuous checks and reviews and changed to the right one. This should in the end contribute to an integrated schema with high quality since an additional check and review has been conducted. For instance, if during comparison of the schemata we have recognized a synonym conflict, e.g., *Customer* in schema 1 and *Client* in schema 2 (see Figure 3), it should be resolved during the conforming of the schemata by introducing a resolution method that is applicable for the current level of abstraction.

It should be noted that if the schemata are designed on an implementation neutral level it is important that a resolution method that retains all concept names and dependencies are applied since they might be of importance for one or several stakeholders. We should therefore not rename one or both concept names, which is one of the most common proposed resolution methods for a synonym conflict but instead introduce a resolution method that keeps both concept names. One way to fulfill this could be to introduce mutual inheritance dependency described as: A and B are synonyms if and only if A inherits B and B inherits A [39].

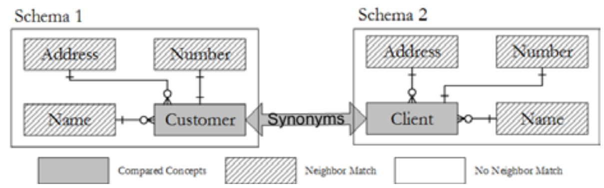

Figure 3. Recognition of synonyms [38] (p. 71)

2) Information architect, stakeholder participation and standards

By involving not only the information architect but also the stakeholders, several of the mentioned pitfalls can be recognized and addressed in the current iteration cycle. This is motivated since it is the stakeholder and the information architect that possess the knowledge about their organization and on how the concepts should be named and connected to each other. The information architect also has to take into account already existing source schemata within the enterprise while a stakeholder might instead focus on integrating a schema of a specific department.

Naming conventions, standards and ontologies, so-called knowledge repositories, might also exist in the enterprise that need to be taken into account in the integration. It should be noted that these naming conventions and standards do not restrict the naming of the concepts which impoverish the language used in the schema but instead are used as a tool to facilitate the integration process as such. Therefore, standards should *not* enforce the usage of one concept name but instead give guidelines on how concept names should be used such as name concepts in the singular.

3) Modeling languages, its external representations and boundary objects

Conventions are not only necessary for the naming of the schema elements. If a language does not have a one to one mapping but a symbol redundancy exits [20], then one and the same symbol has to be chosen in all schemata. Otherwise it confuses the stakeholders.

Some authors, e.g., [40], even eschew the distinction between classes and attributes if possible. The modeling language ORM [40] focuses on the representation of facts. There are no classes and attributes. Instead, object types are related to each other via roles. KCPM [41] has adopted this strategy. This not only helps to be more stable if requirement changes occur, but also has an advantage in schema integration. Problems of structural conflicts can be avoided.

In KCPM, even glossaries were added as an additional means for representing the schema. Such modeling languages and more sophisticated representations of schema elements can be used for both implementation independent and implementation dependent schemata. If the language itself does not provide a glossary representation, it can be introduced as a boundary object. Additionally, ontologies and knowledge repositories can be seen as boundary objects too.

C. Introducing Inter-Schema Properties to Improve and Clarify Dependencies

Another task in pre-integration (phase 1) and comparison of the schema (phase 2) is the recognition of inter-schema properties. An inter-schema property is not really a conflict but it describes a specific dependency (link) between two concepts often referred to as two concepts that are similar but not exactly the same concept. Two of the most common inter-schema properties as described in literature are holonym-meronym dependencies "part-of" (see Figure 4a) and hypernym-hyponym dependencies "is-a" (see Figure 4b).

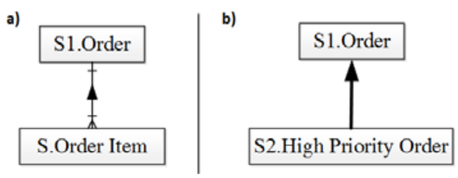

Figure 4. Inter-schema holonym-meronym property (a) and inter-schema hypernym-hyponym property (b)

When two schema elements partly match and have been recognized as an inter-schema property, it is documented and passed to the following phase in the schema integration process, in which it is used as a knowledge repository and/or guideline on how to resolve the partly recognized match.

Introducing and being able to use inter-schema properties in the schema integration process is of great importance since an inter-schema property should have a clearer meaning then, for instance, an association dependency between two concepts. The inter-schema property should therefore also be used not only to clarify and improve a specific meaning of two concepts but also to reduce the number of concepts in the integrated schema if possible. Nevertheless, it should be noted that reducing the number of concepts should be done very carefully. Deleting a concept might not only reduce the quality of the integrated schema but also at the worst violate the completeness quality factor addressed in [24].

Finally, a holonym-meronym dependency might be of two types: aggregation and composition, in which composition is the stronger one.

1) Continuous Checks and Reviews

In the second phase of the schema integration process, comparison of the schemata, either the binary strategy (see Figure 2) or n-ary iterative strategy should be used while recognizing similarities and differences, e.g., inter-schema properties, between two source schemata. When an inter-schema property has been recognized, it should be documented and passed on to the following phases in the integration process. In the end, the inter-schema property should not only, be treated as a source to semantic improvement but also be used as guidance anda knowledge repository.

Nevertheless, an inter-schema property should be used in the right way and not in a way that pollutes the source schemata and/or the integrated schema. In the worst case, an inter-schema property is used in a wrong way causing semantic errors. Applying the best practices of continuous checks and reviews is therefore of great importance to improve not only the quality of the integrated schema as such but also to verify that the inter-schema property is used in a correct way.

For instance, if we in the comparison of the schemata have recognized not only a hypernym-hyponym dependency between concept *Article* and *Product* in schema 1 but also a hypernym-hyponym dependency between concept *Product* and *Article* in schema 2, problems might later on be introduced into the integrated schema (see Figure 5).

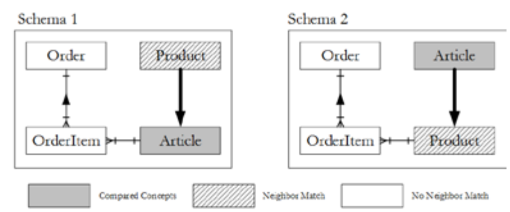

Figure 5. Recognition of difference between two source schemata including inter-schema hypernym-hyponym properties [38] (p. 90)

The inter-schema dependencies are documented and passed on to the following phase, conforming the schemata, in which the schemata are adjusted to solve the recognized conflicts and inter-schema properties. Finally, the modified source schemata (and some extra information resources) are passed to the last phase, merging and restructuring, in which the schemata are first merged and later on restructured. In the worst case, both hypernym-hyponym dependencies described above, *Article* and *Product*, are introduced to the integrated schema causing what is sometimes called reverse subset relationship [21] or cyclic generalization [29] (see Figure 6).

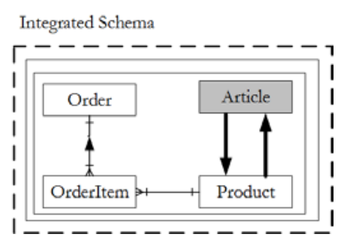

Figure 6. Reverse subset relationship / cyclic generalization (adapted and modified from [38])

However, applying the best practice of continuous checks and reviews, this problem should be recognized and resolved in the current iteration cycle and not left till later iterations in the integration process. Figure 7 illustrates how the reverse subset relationship / cyclic generalization can be resolved by introducing mutual inheritance dependency [39].

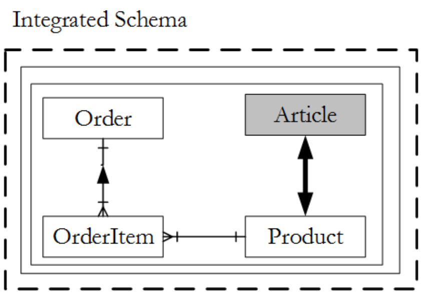

Figure 7. Reverse subset relationship / cyclic generalization resolved by introducing mutual inheritance depencency [38] (p. 90)

2) Information architect, stakeholder participation and standards

Introducing inter-schema properties should result in a semantically richer schema since the inter-schema properties should have a much clearer meaning compared with the association dependency with or without specified cardinality, for instance. Nevertheless, introducing new schema constituents could result in new problems and errors not only since it is the stakeholders that have to be trained in using the new constituent in a correct way, for instance, during a modeling sessions, but also since the new constituent needs to be taken into account during schema integration. Involving information architect as well as stakeholders is also of great importance, since these actors possess the knowledge about their specific domain. Therefore, they also know how to name concepts and how concepts should be connected.

Finally, so-called knowledge repositories (e.g., naming conventions, standards and ontologies) might also exist within the enterprise where the integration is taking place. Ontology, or even domain ontology, might be useful when deciding how to resolve the cyclic generalization dependency. This is the case since a description on how concept *Article* and concept *Product* are dependent might be stated in the ontology (see Figures 5-7).

3) Modeling language and boundary objects

If the modeling language does not provide the possibility to model inter-schema properties, then these dependencies can be seen as a boundary object.

D. Combining Methods, Approaches and Guidelines to Facilitate Recognition of Conflicts

In the first phase, pre-integration, as well as in the second phase, comparison of the schemata, the source schemata are analyzed aiming to recognize similarities and differences within one source schema and between two source schemata, generally referred to as conflicts. In doing so, several matching approaches are needed. The result from each matching approach also needs to be combined into one result. This is motivated since combining the result from several matching approaches into one result should produce a better result than just using the result from one single approach [42]. For instance, in [43], the author has described and exemplified the use of matching approaches for recognizing similarities and differences while integrating structural Karlstad Enterprise Modeling schemata. In [43], the author uses a composite schema based matching approach, in which "[…] the match result of a first matcher is consumed and extended by a second matcher […]" [42] (p. 343) The composite schema based matching approach described in [43] is divided into two parts stating with element level matching followed by structural level matching (Figure 8). Element level matching includes the usage of *concept name comparison*, *linguistic rules* and if a domain ontology exists also *domain ontology-based matching*. Structural level matching includes the usage of *rule-based comparison* and if they exist also *domain ontology matching* and/or *taxonomy based matching*.

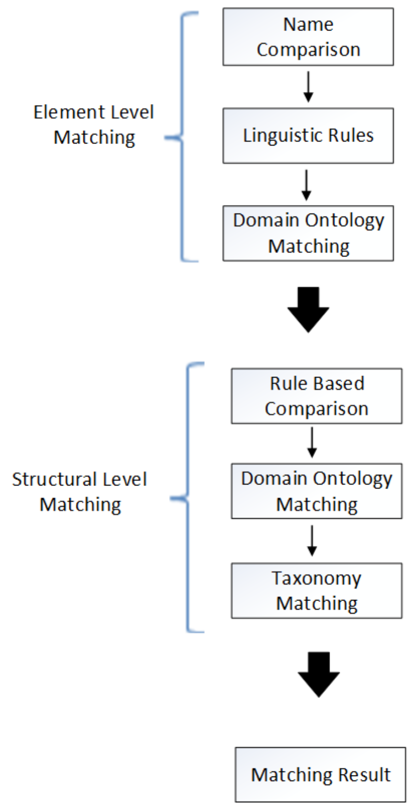

Figure 8. Matching as described and discussed in [43]

To illustrate the importance of using several methods, approaches and guidelines, so-called matchers, we will shortly address some aspects of the matching approach illustrated in Figure 8. In doing so, we use the example schemata illustrated in Figure 9. For a more complete description and discussion of the example, the reader should refer to [43]. It should be noted that in the matching example described below we focus on recognizing similarities and differences between two source schemata. In other words, we emphasise the second phase, comparison of the schemata, mentioned as a challenge [33], also referred to as an important [29] and difficult [31] [32] [44] phase of schema integration. It should also be noted that the end result of the matching approach might include redundant dependencies and concepts since we have not yet decided on what dependency to use, e.g., synonym or inter-schema hypernym-hyponym.

Figure 9a illustrates the result after conducting the first phase, pre-integration and Figure 9b the result after applying the composite schema based matching approach as described in [43] but before deciding if it is a conflict, synonym, or an inter-schema hypernym-hyponym dependency between Order and High Priority Order. Finally, Figure 9c shows the legend of the used symbols in Figure 9.

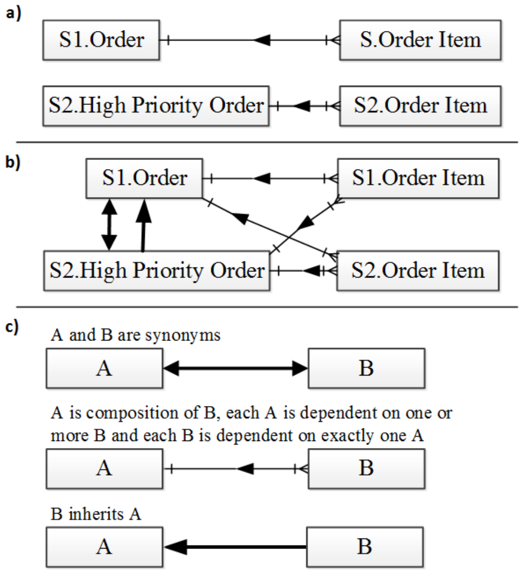

Figure 9. Illustrating example of matching approaches (adapted and modifed from [43])

Figure 9a should be interpreted as follows: *S1.Order* is composed of one or several *S1.Order Item* while *S1.Order Item* is part-of (dependent on) exactly one *S1.Order*. The same interpretation is applicable for *S2* with the change of concept name from *S1.Order* to *S2.High Priority Order*.

Figure 9b should be interpreted as follows: *S1.Order* is composed of one or several *S1.Order Item* and *S2.Order Item* while *S1.Order Item* and *S2.Order item* is part-of exactly one *S1.Order*. *S2.High Priority Order* is composed of one or several *S1.Order Item* and *S2.Order Item* while *S1.Order Item* and *S2.Order item* is part-of exactly one *S2.High Priority Order*. Finally, *S2.High Priority Order* is-a *S1.Order* and *S1.Order* and *S2.High Priority Order* are synonyms.

Focusing on name comparison, linguistic rules and rule based comparison, we may describe the process of matching the two source schemata in Figure 9a as follows:

Name comparison, the labels of schema 1 are compared to the labels of schema 2, on the element level results in the following correspondences: *S1.Order Item = S2.Order Item, S1.Order Item ~ S2.High Priority Order, S1.Order ~ S2.Order Item* and *S1.Order ~ S2.High Priority Order*.

Applying two linguistic rules on the element level sharpen the meaning of the three last correspondences as follows: *S1.Order Item belongs/related to S2.High Priority Order, S2.Order Item belongs/related to S1.Order* and *S2.High Priority Order is-a S1.Order*. The linguistic rules are in [30] (p. 415) described as follows:

a) *If the compared schema elements have names in the form of A and AB […], then the relationship* ***"AB belongs/related to A"*** *can be assumed between the elements.*

b) *If the compared schema elements have names in the form B and AB […], then the relationship* ***"AB is a B"*** *can be assumed between the elements.*

Applying rule based comparison, first addressed in [38] and later adapted and modified in [30] [43] [45] [46], on the structural level, the following might be suggested: *S1.Order is synonymic with S2.High Priority Order* (1), *S2.High Priority Order is-a S1.Order* (2), *S2.Order Item is part-of (composition) S1.Order* (3), *S1.Order Item is part-of (composition) S2.High Priority Order* (3) and finally *S2.High Priority Order is-a S1.Order* (4).

The rules applied in the presented example might be described as follows:

1) If the comparison of concept names, element level matching, yields match and the comparison of concepts neighborhood, structural level matching, yields partial match, with one concept in each source schemata named differently, then synonymic concepts are most likely recognized.
2) If the comparison of concept names, element level matching, yields match and the comparison of concepts neighborhood, structural level matching, yields partial match, with one concept name named with prior addition to the other one, then an inter-schema hypernym-hyponym property is most likely recognized.
3) If the comparison of concept names, element level matching, yields partially match and the comparison of concepts neighborhood, structural level matching, yields partial match or match, with one concept named with a following addition to the other one, then an inter-schema holonym-meronym property is most likely recognized.
4) If the comparison of concept names, element level matching, yields partially match and the comparison of concepts neighborhood, structural level matching, yields partial match or match, with one concept named with a prior addition to the other one, then an inter-schema hypernym-hyponym property is most likely recognized.

1) *Continuous Checks and Reviews*

Having designed the source schemata, it is important that in both pre-integration and comparison of the schemata a combination of matching methods, approaches and guidelines are used to recognize not only conflicts within one source schema but also conflicts between two source schemata. In doing so, it is possible to check the quality of the schema after each matching approach has been applied and if necessary also to review the schema. It should, however, be noted that the schemata produced using each matching method, approach and guideline are intermediate versions of the schemata that are finally going to be integrated. For instance, in the example described above, name comparison results in several matches while the linguistic rules that follow sharpen the meaning of these matches ending up with new intermediate versions of the schemata. Doing continuous checks and reviews after each matching method, approach and guideline has been applied, should contribute to a high quality integrated schema. This is motivated since recognizing problems as early as possible should contribute to a review that is not as cumbersome as identifying problems later on in the integration process resulting in big changes.

2) Information Architect, Stakeholder Participation and Standards

As addressed in [42], the selection of matchers, in our case methods, approaches and guidelines can be made both automatically and manually by a user. However, a generic automated solution process, which selects methods, approaches and guidelines to combine, is difficult to accomplish and besides, a manual selection process is easier to implement. A semi-automatic approach, including both automatic and manual tasks, should therefore be chosen. During such semi-automatic approach, the information architect as well as stakeholders should be very much involved. This is also emphasized in [42], in which the authors state that "[…] user interaction is necessary in any case […]" (p. 343), referring to the process of selection of matchers. This is also in line with [24], who state: "Involvement of all stakeholders in the data modelling process was found to be more important than any other single issue in achieving quality improvements." (p. 646). In general, the information architect and the stakeholders should be involved during the whole integration process meaning that they should also be involved while checking and reviewing each source schemata after each method, approach and guideline has been applied.

Finally, standards such as naming conventions are also important to take into consideration during schema matching. This is motivated since ontologies [47] as well as lexicons such as WordNet [48] are useful in the process of recognizing similarities and differences and should therefore be part of schema matching.

E. Restructuring

Restructuring is the last task within the fourth phase (merging and restructuring). If there is a need to semantically enrich the schema, then detected inter-schema properties can be introduced. Usually, the better the steps and tasks before have been executed, the less has to be done for restructuring.
However, especially for the implementation dependent level, restructuring is needed to prune and optimize the resulting schema. It is also necessary if schema alignment and merging were done automatically. Once again a semantic and pragmatic understanding of the terms is required.

1) Continuous checks and reviews

Continuous checks and reviews can be applied as a quality improving instrument to avoid that pruning leads to a schema that is not any longer agreed on by all involved parties. After each major restructuring solution the schema should be checked if its semantic content has not been lost.

2) Information architect

Once again the information architect can act as a mediator between the stakeholders. If necessary, it is his obligation to describe the pragmatics and effects of a schema change. He is also a supervisor for executing several schema checking and restructuring methods and as an organizational interface when working with boundary objects.

3) Stakeholder participation

The more stakeholders from different interest group are involved the more perspectives are considered.

4) Modeling languages and boundary objects

Good modeling languages or additional boundary objects can prevent misunderstandings and errors.

In [49], the author has proposed the introduction of icons and pictures to improve the comprehensibility of conceptual schemata. Others [40] [50] have focused on verbalization of conceptual schemata. Verbalization is a technique, where elements of a schema are translated back to its natural language representation. For a class diagram this means that classes and attributes are mainly translated into nouns and noun phrases. Associations between classes are translated into sentences, which contain the translated classes as the sentence subject and objects. They argued that a conceptual schema, which is transformed back to natural languages sentences representing facts of the domain, is more understandable. Especially, non-computer scientists, who are not familiar with the notions and notations of a modeling language, will be supported with this approach.

For implementation dependent schemata additional techniques adopted from model checking and validation can be used for restructuring. Suppose the schema was developed for an information system, particularly for the database used by the information system. Forms (i.e., user interfaces) can help the stakeholders to understand, which schema elements are necessary for which features of the information system.

Visual query languages [51], which allow the navigation through the conceptual schema, are another way of checking the schema. For the OMT modeling language, [52] has proposed the manual checking of the schema against natural language queries. In [53], a proposal was made on how this can be automated. Advantages and technical problems were discussed. With visual or controlled natural language query languages, manual query checking and/or systems that check the schema automatically based on queries, a restructuring and validation cycle can work as illustrated in Figure 10.

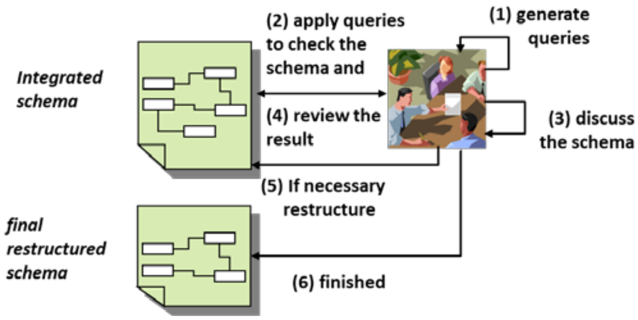

Figure 10. Restructuring task with queries

At the beginning, there is the integrated schema. Since the schema will be implementation dependent, meaning that there will a database working on base of the schema, queries should be generated (1). These are the queries, which most likely will be applied on the integrated database schema. The Information architect together with the stakeholders should generate these queries. Afterwards the queries are executed manually or automatically with a tool on the conceptual schema (2). The result of these executions should be reviewed (2) and the schema should be discussed (3), i.e., is the database schema already optimized for the queries or not. Depending on the result (4), the schema should be restructured and optimized (5) or it turns out that the schema already fits with the retrieval requirements of the stakeholders (6). In this case, the final restructured schema was developed. Finally, with a modeling language (e.g., ORM or KCPM) that does not distinguish between classes and attributes but nevertheless provides mappings to the logical database schema restructuring can be supported. There is no need to prune a class to an attribute if this class does not have further properties. The transformation rules given by these modeling languages will do this.

These examples show that choosing the right modeling language does not only help during the modeling but also later supports possible schema integration. If the original modeling language used is not sufficient, all the stakeholders (i.e., the information architect and the other involved stakeholders) should agree on some boundary objects (e.g., glossaries, or methods such as querying) to get a common understanding and negotiate about the schema.

VI. Conclusion and Future Work

In this article, we have focused on schema quality within the schema integration process. In doing so, we have addressed five best practices of quality improvement given in the literature and five specific integration tasks that should increase the quality of the schema being designed. The best practices addressed are: continuous checks and reviews, information architect, stakeholder participation and standards, a good modeling language as well as the use of boundary objects. The five integration tasks addressed are: choosing the right integration strategy, choosing the right conflict resolution methods for the chosen level of abstraction, introducing inter-schema properties to improve and clarify dependencies, combining methods and approaches and guidelines to conflict resolution and finally restructuring. Wherever it is possible, we have also addressed, within each integration task, how the best practices might be used to aid in the process of producing a high quality schema.

To conclude (see also Table II), the best practices used for conceptual modeling if addressed in connection to schema integration can improve the five mentioned tasks and hence the integration process as such. Continuous checks and reviews, information architect and stakeholder participation can be drivers for choosing the right integration strategy. Continuous checks and reviews, standards, information architect and stakeholder participation are essential in the conflict resolution task. The more conflicts are checked and resolved the better. The more the stakeholders and the information architect is involved the more conflicts can be resolved. Standards support this task as long as they do not restrict the specific naming of enterprise concepts.

For the inter schema property introduction, which is used in at least two phases of the integration process, continuous checks and reviews can help to verify that the inter-schema property is used in the correct way. Stakeholders and the information architect are those who possess the domain knowledge and can thus support the aim to get a semantically richer schema with clear meanings. Standards and ontologies are useful to support the detection of inter-schema properties.

Modeling language of good quality (i.e., language with specific features and a good external representation that supports schema understanding) is important for the following task: choosing the right conflict resolution methods for the chosen level of abstraction, introducing inter-schema properties to improve and clarify dependencies and during restricting. Boundary objects can also be applied in choosing the right conflict resolution methods for the chosen level of abstraction, introducing inter-schema properties to improve and clarify dependencies and during restricting.

In the long run, these improved tasks contribute to a high quality integrated schema.

To summarize, in this work we have described a framework for the schema integration process and aligned best practices to task. In future, we will study some of the best practices in detail. For instance, the kinds of external representations that are good for the schema integration purpose are important to identify. This question is also important for boundary objects. Are there specific boundary objects that should be preferred? It might also be important to determine if the approaches to restructuring the implementation dependent schema can also be applied to implementation independent schemata.

TABLE II. BEST PRACTICES AND INTEGRATION TASKS

Integration tasks/Best Practice	Choosing the Right Integration Strategy	Choosing the Right Conflict Resolution Methods for the Chosen Level of Abstraction	Introducing Inter-Schema Properties to Improve and Clarify Dependencies	Combining Methods, Approaches and Guidelines to Facilitate Recognition of Conflicts	Restructuring
Continuous Checks and Reviews	Are facilitated by the ladder, balanced and iterative integration strategy.	Are the enablers to verify that the schemata illustrate the chosen level of abstraction during the whole integration process.	Are the enablers to verify that the inter-schema properties are used in a correct way during the whole integration process.	Are the enablers to verify that each method, approach and guideline are used in a correct way and that each of these contributes to a more reliable matching result during the whole integration process.	Are the enablers to verify that the schemata are correct after each greater restructuring.
Information Architect	Checks and verifies, from the perspective of the information architect, that an appropriate integration strategy is choosen.	Checks that the choosen conflict resolution methods are in compliance with existing enterprise schemata.	Checks that the introduced inter-schema properties are in compliance with existing enterprise schemata.	Checks that the result from each method, approach, and guideline complies with existing enterprise schemata.	Moderates the restructruing process. The information architect can give information on which pragmatic effects a certain restructuring decision can have.
Stakeholder Participation	Checks and verifies, from the perspective of the stakeholders, that an appropriate integration strategy is choosen.	Checks that choosen conflict resoluton methods are semantically correct and that the schema is complete from a stakeholder perspective.	Checks that the introduced inter-schema properties are semantically correct and that the schema is complete from a stakeholder perspective.	Checks that each method, approach and guideline produces semantically correct results and that the schema is complete from a stakeholder perspective.	Improves the schema quality since the stakeholders are informed about the effects of a restructing decision and therefore also can influence how restructuring is performed.
Standards	Help in the process of checking that the schemata are syntactically correct and that terms are used in compliance with the enterprise schemata.	Help in the process of introducing the correct resolution method for not only naming conflicts but also structural conflicts.	Help in the process of introducing the correct inter-schema proptery and help in the process of introducing the inter-schema property in a correct way.	Help in the process of introducing and applying each method, approach and guideline.	Not applicable
Modeling language quality	Not applicable	A modeling language, which does not distinguish between classes and attributes can prevent structural conflicts	Not applicable	Not applicable	The modeling language and its external representation can support restructuring (e.g., by providing modeling elements that make restructuring easier)
Boundary objects	Not applicable	With the right boundary objects, the stakeholders can be made aware that conflicts exist.	Interschema properties can be seen as boundary objects.	Not applicable	Boundary objects in the form of additional methods can help to identify possible errors and can support the understanding of the schema.

REFERENCES

[1] P. Bellström and C. Kop, “Towards Quality Driven Schema Integration Process Tasks,” Proceedings of the The Sixth International Conference on Information, Process, and Knowledge Management (eKNOW 2014), 2014, pp. 98-104.

[2] C. Batini and M. Lenzerini, “A Methodology for Data Schema Integration in the Entity-Relationship Model,” IEEE Transactions on Software Engineering, vol. 10 (6), 1984, pp. 650-664.

[3] C. Batini, M. Lenzerini, and S. B. Navathe, “A Compartive Analysis of Methodologies for Database Schema Integration,” ACM Computing Surveys, 1986, vol. 18 (4), pp. 323-364.

[4] J. A. Larson, S. B. Navathe, and R. Elmasri, “A Theory of Attribute Equivalence in Databases with Application to Schema Integration,” Transactions on Software Engineering, vol. 15 (4), 1989, pp. 449-463.

[5] A. Savasere, A. Sheth, and S. Gala, “On Applying Classification to Schema Integration,” Proceedings of the First International Workshop on Interoperability in

Multidatabase Systems (IMS'91), IEEE Press, 1991, pp. 258-261.

[6] P. Johannesson, "A Logical Basis for Schema Integration," Third International Workshop on Research Issues on Data Engineering (RIDE-IMS'93), IEEE Press, 1993, pp. 86-95.

[7] H. K. Bhargava and R. M. Beyer, "Automated Detection of Naming Conflicts in Schema Integration: Experiments with Quiddities," Proceedings of the 25th Hawaii International Conference on System Sciences, IEEE Press, 1992, pp. 300-310.

[8] J. Geller, A. Mehta, Y. Perl, E. Neuhold, and A. Sheth, "Algorithms for Structural Schema Integration," Proceedings of the Second International Conference on Systems Integration (ICSI'92), IEEE Press, 1992, pp. 604-614.

[9] M. García-Solaco, F. Saltor, and M. Castellanos, "A Structure Based Schema Integration Methodology," Proceedings of the Eleventh International Conference on Data Engineering, IEEE Press, 1995, pp. 505-512.

[10] H. Dai, "An Object-Oriented Approach to Schema Integration and Data Mining in Multiple Databases," Proceedings on the Technology of Object-Oriented Languages (TOOLS), IEEE Press, 1997, pp. 294-303.

[11] E. Métais, Z. Kedad, I. Comyn-Wattiau, and M. Bouzeghoub, "Using Linguistic Knowledge in View Integration: Toward a Third Generation of Tools," Data & Knowledge Engineering, vol. 23 (1), 1997, pp. 59-78.

[12] S. Ram and V. Ramesh, "A Blackboard-Based Cooperative System for Schema Integration," IEEE Expert, 1995, vol. 10 (3), pp. 56-62.

[13] S. Spaccapietra and C. Parent, "View Integration: a Step Forward in Solving Structural Conflicts," IEEE Transactions on Knowledge and Data Engineering, vol. 6 (2), 1994, pp. 258-274.

[14] H. Frank and J. Eder, "Towards an Automatic Integration of Statecharts," International Conference on Conceptual Modeling (ER 1999), 1999, pp. 430-444.

[15] B. H. C. Cheng and E. Y. Wang, "Formalizing and Integrating the Dynamic Model for Object Oriented Modeling," IEEE Transactions on Software Engineering, vol. 28 (8), 2002, pp. 747-762.

[16] M. Stumptner, M. Schrefl, and G. Grossmann, "On the Road to Behavior-Based Integration," Proceedings of the 1st APCCM Conference, 2004, pp. 15-22.

[17] A. Raut, "Enterprise Business Process Integration," Conference on Convergent Technologies for Asia-Pacific Region, IEEE Press, 2003, pp. 1549-1553.

[18] S. Fan, L. Zhang, and Z. Sung, "An Ontology Based Method for Business Process Integration," International Conference on Interoperability for Enterprise Software and Applications in China, IEEE Press, 2008, pp. 135-139.

[19] W. J. Lee, S. D. Cha, and Y. R. Kwon, "Integration and Analysis of Use Cases Using Modular Petri Nets in Requirements Engineering," IEEE Transaction of Software Engineering, vol. 24 (12), 1998, pp. 1115-1130.

[20] K. Winter, I. J. Hayes, and R. Colvin, "Integrating Requirements: The Behavior Tree Philosophy," 8th IEEE International Conference on Conference on Software Engineering and Formal Methods (SEFM), IEEE Press, 2010, pp.41-50.

[21] C. Batini, S. Ceri, and S. B. Navathe, Conceptual Database Design an Entity Relationship Approach. Redwood City: Benjamin/Cummings Publishing Company, 1992.

[22] O. L. Lindland, G. Sindre, and A. Solvberg, "Understanding Quality in Conceptual Modeling," IEEE Software, vol. 11 (2), 1994, pp. 42-49.

[23] D. L. Moody, "Theoretical and Practical Issues in Evaluating the Quality of Conceptual Models: Current State and Future Directions," Data & Knowledge Engineering, vol. 55 (3), 2005, pp. 243-276.

[24] D. L. Moody and G. G. Shanks, "Improving the Quality of Data Models: Empirical Validation of a Quality Management Framework," Information Systems Journal, vol. 28 (2), 2003, pp. 619-650.

[25] S. S. Cherfi, J. Akoka, and I. Comyn-Wattiau, "Perceived vs. Measured Quality of Conceptual Schemas: An Experimental Comparision," Proceedings of Tutorials, Posters, Panels and Industrial Contribution of the Twenty-Sixth International Conference on Conceptual Modeling (ER 2007), vol. 83, 2007, pp. 185-190.

[26] J. Becker, M. Rosemann and C. von Uthman, "Guidelines of Business Process Modeling," Business Process Management, LNCS 1806, 2000, pp. 30-49.

[27] J. Krogstie, Model based Development and Evolution of Information Systems – A Quality Approach. London: Springer, 2012.

[28] P. Bellström, View Integration in Conceptual Database Design Problems, Approaches and Solutions. Licentiate Thesis. Karlstad Univeristy, Karlstad University Press 2006:5, 2006.

[29] W. Song, Schema Integration – Principles, Methods, and Applications. Dissertation. Stockholm: Stockholm University & The Royal Institute of Technology No. 95-019, 1995.

[30] P. Bellström and J. Vöhringer, "A Semi-Automatic Method for Matching Schema Elements in the Integration of Structural Pre-Design Schemata," International Journal on Advances in Intelligent Systems, vol. 4 (3 & 4), 2011, pp. 410-422.

[31] L. Ekenberg and P. Johannesson, "A Formal Basis for Dynamic Schema Integration," Conceptual Modeling – (ER'96), LNCS 1157, 1996, pp. 211-226.

[32] L. Lee and T. W. Ling, "A Methodology for Structural Conflict Resolution in the Integration of Entity-Relationship Schemas," Knowledge and Information Systems, vol. 5 (2), 2003, pp. 225-247.

[33] P. Johannesson, Schema Integration, Schema Translation, and Interoperability in Federated Information Systems. Dissertation. Stockholm University & Royal Institute of Technology No. 93-010-DSV, 1993.

[34] S. L. Star and J. R., Griesemer, "Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkley's Museum of Vertebrate Zoology, ", 1907-39, Social Studies of Science, 19 (3), 1989, pp. 387-420.

[35] P. R. Carlile, "Transferring, Translating and Transforming. An Integrative Framework for Managing Knowledge Across Boundaries," Organization Science, 15 (5), 2004, pp. 555-568.

[36] R. Abraham, "Enterprise Architecture Artifacts As Boundary Objects – A Framework of Properties," Proceedings of the 21st European Conference on Information Systems (ECIS), 2013, Paper 120.

[37] M. Boman, Jr. J. A. Bubenko, P. Johannesson, and B. Wangler, Conceptual Modelling. London: Prentice Hall, 1997.

[38] P. Bellström, Schema Integration How to Integrate Static and Dynamic Database Schemata. Dissertation. Karlstad University, Karlstad University Studies 2010:13, 2010.

[39] R. Gustas, Semantic and Pragmatic Dependencies of Information Systems. Kaunas: Kaunas Technologija, 1997.

[40] T. Halpin and M. Curland, "Automated Verbalization for ORM-2," Proceedings of OTM 2006 Workshop , LNCS Vol 4278, 2006, pp. 1181-1190.

[41] H.C. Mayr and C. Kop, "Conceptual Predeign – Bridging the Gap between Requirements and Conceptual Design,"

Proceedings of the 3rd Int. Conference on Requirements Engineering (ICRE'98), IEEE Press, 1998, pp. 90-100.

[42] E. Rahm and P.A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," The VLDB Journal, vol. 10, 2001, pp. 334-350.

[43] P. Bellström, "A Semi-Automatic Approach for the Integration of Structural Karlstad Enterprise Modeling Schemata," Advances in The Human Side of Service Engineering, 2014, pp. 13-24.

[44] A. Doan, F. N. Noy and A. Y. Halevy, "Introduction to the Special Issue on Semantic Integration," SIGMOD Record, vol 33 (4), 2004, pp. 11-13.

[45] P. Bellström, "A Rule-Based Approach for the Recognition of Similarities and Differences in the Integration of Structural Karlstad Enterprise Modeling Schemata," The Practice of Enterprise Modeling, 2010, pp. 177-189.

[46] P. Bellström and J. Vöhringer, "A Three-Tier Matching Strategy for Predesign Schema Elements," The Third International Conference on Information, Process, and Knowledge Management (eKNOW 2011), 2011, pp. 24-29.

[47] T.R. Gruber, "A Translation Approach to Portable Ontology Specifications," Knowledge Acquisition, 5, 1993, pp. 199-220.

[48] G.A. Miller, "WordNet: A Lexical Database for English," Communication of the ACM, 38 (11), 1995, pp. 39-41.

[49] D. Moody, "The 'Physics' of Notations: Toward a ScientificBasis for Constructing Visual Notations in Software Engineering," IEEE Transaction on Software Engineering, vol. 35 (6), 2009, pp. 756-779.

[50] H. Dalianis, "A Method for Validating a Conceptual Model by Natural Language Discourse Generation," Proceedings of the Fourth International Conference CAiSE'92 on Advanced Information Systems Engineering, LNCS Vol 594, 1992, pp. 425-444.

[51] K. Järvelin, T. Niemi, A. Salminen, "The Visual Query Language CQL for Transitive and Relational Computation," Data & Knowledge Engineering, vol. 35, 2000, pp. 39-51.

[52] J. Rumbaugh, M. Blaha, W. Premelani, F. Eddy and W. Lorensen, Object oriented Modeling and Design. Prentice Hall International Inc. Publ. Comp. 1991.

[53] C. Kop, "Checking Feasible Completeness of Domain Models with Natural Language Queries," Proceedings of the 8th Asia-Pacific Conference on Conceptual Modeling, vol. 130, 2012, pp. 33- 42.