Formalizing Linguistic Conventions for Conceptual Models

Jörg Becker, Patrick Delfmann, Sebastian Herwig, Łukasz Lis, and Armin Stein

University of Münster, European Research Center for Information Systems (ERCIS), Leonardo-Campus 3, 48149 Münster, Germany
{becker,delfmann,herwig,lis,stein}@ercis.uni-muenster.de

Abstract. A precondition for the appropriate analysis of conceptual models is not only their syntactic correctness but also their semantic comparability. Assuring comparability is challenging especially when models are developed by different persons. Empirical studies show that such models can vary heavily, especially in model element naming, even if they express the same issue. In contrast to most ontology-driven approaches proposing the resolution of these differences ex-post, we introduce an approach that avoids naming differences in conceptual models already during modeling. Therefore we formalize naming conventions combining domain thesauri and phrase structures based on a linguistic grammar. This allows for guiding modelers automatically during the modeling process using standardized labels for model elements. Our approach is generic, making it applicable for any modeling language.

Keywords: Conceptual Modeling, Naming Conventions, Linguistics.

1 Introduction

Empirical studies show that especially those conceptual models which are being developed in a timely and regionally distributed way can vary heavily concerning terms and structure. Thus, so-called *naming conflicts* and *structural conflicts* [1, 2] may occur, even if the same issue is addressed [3]. Moreover, even models of the same issue developed by the same persons at different times may show intense variations. Consequently, the analysis of conceptual models – for example for integration or benchmarking purposes – may be extremely laborious. Information that is expressed in different ways has to be "standardized" in some way in order to make the models comparable. Usually, an according standardization process requires discussions including all involved modelers in order to reach a consensus. Sometimes, even external consultants are involved additionally [4, 5].

In order to solve this problem, approaches are required that are able to assure model comparability. In the literature, there exist many contributions that propose approaches for resolving modeling conflicts in conceptual models subsequent to modeling (cf. Section 2). Unlike these approaches, the goal of this article is to introduce an approach that ensures the comparability of conceptual models by avoiding potential conflicts already *during* modeling. This way, we prevent problems that result from the ex-post resolution of conflicts and make the standardization process described above dispensable. This article focuses on *naming conflicts*. We define naming conventions for elements of modeling languages and ensure their compliance by an *automated, methodical guiding* during modeling. The conventions are set up using *domain terms* and *phrase structures* that are defined as valid in the regarded modeling context. As a formal specification basis, we use thesauri that provide term conventions not only for nouns but also for verbs and adjectives, including descriptions of their meanings. In order to provide conventions for phrase structures, we make use of a linguistic specification approach. During modeling, model element names are validated simultaneously against both the term and phrase structure conventions. Our approach is generic so that it can be applied to any conceptual modeling language. The approach is suitable for modeling situations, where it is possible to provide all involved stakeholders with the necessary information about the modeling conventions, meaning modeling projects that are determined regarding organization and/or business domain. These modeling situations usually occur in companies, corporate groups, or modeling communities.

This paper is structured as follows. First, we analyze related work on naming conflict resolution in Section 2 and discuss the research gap that led to the development of the approach presented in this paper. Since process model elements are usually named with complex phrases rather than with single terms, they are extra prone to naming conflicts. An explorative analysis of naming practice in process models shows the potential of our approach in the case of process modeling. Furthermore, we outline our research methodology. In Section 3, we introduce a conceptual framework for the specification and enforcement of naming conventions. The feasibility of our approach is shown exemplarily with a demonstrator software in Section 4. We conclude the paper in Section 5 and motivate further research.

2 Foundations

2.1 Related Work

Early approaches of the 1980s and 1990s discussing the resolution of naming conflicts address the integration of company databases and use the underlying *schemas* as a starting point [1, 2, 6, 7]. Hence, these approaches focus on data modeling languages, mostly dialects of the Entity-Relationship Model (ERM) [8]. Names of schema elements are compared, and this way, similarities are revealed. The authors state that such a semantic comparison can exclusively happen manually. Moreover, only single nouns are considered as names. In contrast, in common conceptual modeling languages (especially process modeling languages), names are used that consist of sentence fragments containing terms of any word class. Thus, these early approaches are only suitable for data modeling languages as a specific class of conceptual modeling languages.

Other approaches make use of *ontologies* [9, 10] in order to address the problem of semantic comparison of names. Those approaches can be distinguished into two different kinds. On the one hand, authors act under the assumption that there exists a “generally accepted” ontology describing a certain modeling domain. It is assumed that all considered models of this domain comply with its ontology, meaning that modelers had a thorough knowledge of the ontology *before* the modeling took place. On the other hand, approaches suggest deriving an ontology from the models that have to be analyzed, which has to be performed *after* the modeling took place.

There are a few examples for the former approach. For example, [11] propose adopting terms from existing ontologies for process models manually. But due to manual adoption, correctness cannot be assured. [12] propose semi-automated adoption of model element names. However, they restrict their approach to BPMN models [13]. Furthermore, only the fact that two modelers act in the same business domain does not guarantee that they share the same or an equivalent understanding of business terms. If a "generally accepted" ontology is available, it is suitable for model comparison if and only if it is explicated and can be accessed by all involved modelers already *during* the modeling process. Additionally, in order to ensure comparability of the models, modelers have to comply strictly with the ontology. Most approaches make the implicit assumption that these preconditions are already given rather than addressing a methodical support.

For the latter approach, [14] connects domain ontologies to the terms that are used as names in conceptual models. This way, he establishes relationships between elements of different models that are to be analyzed. In addition to ontologies, [15] define combined similarity measures that consist of syntactic and semantic parts. These serve as a basis for the decision whether the model elements compared are equivalent or not. Consequently, it is argued that if identical terms – or those that are defined as synonymous within the ontology – are used in different models and by different modelers, these can be considered as semantically identical as well [16, 17]. It has to be questioned whether the advantage of the subsequent connection of the models via the ontology warrants the efforts in comparison to a conventional manual analysis.

Only a few approaches, mainly originating from the German speaking area, suggest standardized phrases for model element names in order to increase the clarity of process models. For example, [18] and [19] propose particular phrase structure guidelines for names of process activities (e.g., *<verb, imperative>* *<noun, singular>*; in particular "check invoice"). Moreover, the authors propose so-called *Technical Term Models* [20] that have to be created previously to process modeling and that specify the terms to be used within the phrases. However, the scope of Technical Term Models is restricted to nouns. Similar approaches provided by [16] and [17] propose the provision of generally accepted vocabularies. Further approaches recommend connecting names of model elements to online dictionaries (e.g., [21]) in order to establish semantic relationships of terms [22, 23]. These online dictionaries consist of extensive collections of English nouns, verbs, and adjectives as well as their semantic relationships. Actually, the proposed approaches are promising regarding increased comparability of conceptual models since all of them aim at standardizing names for model elements *prior to modeling*. However, up to now, a methodical realization is missing.

To sum up, we identify the following need for development towards avoiding naming conflicts in conceptual models: Up to now, methodical support for (1) the formal specification of naming conventions for all word classes and (2) the formal specification of phrase structure conventions is missing. Furthermore, there exists no methodical support for (3) guiding modelers in order to *comply* with the conventions.

To realize such a methodical support, we propose an approach that consists of (1) a formalism to specify thesauri covering nouns, verbs, and adjectives, (2) a grammar to specify phrase structures that can hold terms specified as valid within the thesauri, and (3) a procedure model to guide modelers automatically in complying with the conventions.

2.2 Naming Practice in Process Models

Naming practices in process models provide evidence concerning the danger of naming conflicts as well as requirements to approaches that aim at resolving or even avoiding them. Therefore, we conducted an exploratory empirical analysis of two modeling projects consisting of overall 257 Event-driven Process Chain (EPC [24]) models containing in turn overall 3918 elements (1827 functions and 2091 events). Within these modeling projects, modeling conventions were available in terms of glossaries and phrase structures. However, these conventions solely existed as textual recommendations rather than methodical support. All model element names were parsed with *TreeTagger* [25] and revised manually. We found out that, first, most elements were named with complex phrases rather than with single terms (cf. Fig. 1).

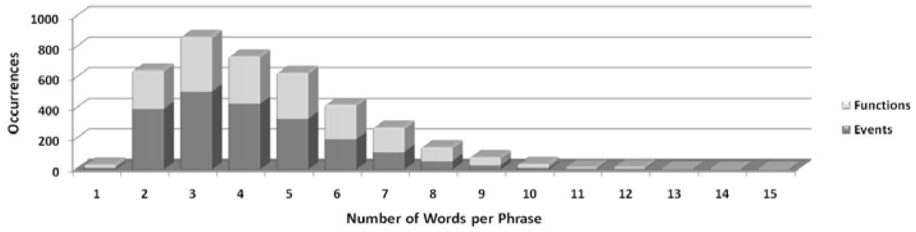


Fig. 1. Average Number of Words Used in Process Model Element Names

Second, element names containing a certain number of terms consisted of many different phrase structures (e.g., *<verb, imperative><noun, singular>*, in particular "audit invoice" or *<noun, singular><verb, gerund>*, "invoice auditing"; cf. Table 1). The results show that process models are extra prone to naming conflicts, since process model elements are usually named with sentence fragments rather than with single terms. Approaches towards resolving or avoiding naming conflicts therefore have to consider not only the terms but also the phrase structures used in model element names.

Table 1. Phrase Structures in Process Model Element Names

# of Terms	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
# of Events	10	396	509	429	331	197	114	55	27	10	4	4	2	2	1
# of Different Phrase Structures (Event)	6	37	136	221	248	175	102	54	26	10	4	4	2	2	1
# of Functions	21	252	358	310	301	225	160	90	52	26	12	13	2	3	2
# of Different Phrase Structures (Function)	3	29	85	157	204	193	141	87	52	25	12	13	2	3	2

2.3 Research Methodology

The research methodology followed here complies with the Design Science approach [26] that deals with the construction of scientific artifacts like methods, languages, models, and implementations. Following the Design Science approach, it is necessary to assure that the research addresses a *relevant problem*. This relevance has to be proven. Furthermore, the artifacts to be constructed have to represent an *innovative contribution* to the existing knowledge base within the actual research discipline. Similar or identical solutions must not be already available. Subsequent to the *construction* of the *artifacts*, these have to be *evaluated* in order to prove their fulfillment of the research goals.

In this contribution the *scientific artifact* is the modeling approach outlined in Section 1. This artifact aims at solving the *relevant problem* of the lacking comparability of conceptual models (cf. Section 1). Related work does not provide satisfactory solutions up to now (cf. Section 2). Hence, the approach presented here (cf. Section 3) makes an *innovative contribution* to the existing knowledge base. In order to *evaluate* the approach, we have implemented a demonstrator software that shows the general applicability of the approach (cf. Section 4). Further evaluations concerning acceptance as well as efficiency and increase of comparability will be the subject of empirical studies to be performed in the short term (cf. Section 5).

3 A Framework for the Specification and Enforcement of Naming Conventions

3.1 Procedure Model

In order to provide a framework for naming conventions, we propose the usage of a specific language that is used for naming model elements in a certain modeling context (i.e., a specific modeling domain, project, or company). This domain language is a subset of the respective natural language (here: English) used in the modeling context. The domain language consists of a set of valid domain terms that are allowed to be used in model element names exclusively. That is, the set of domain terms is a subset of all terms available in the respective natural language. Furthermore, every natural language has a certain syntax that determines the set of grammatically correct phrases. In our framework, we restrict the syntax of the respective natural language as well. This means that the possibilities to construct sentences for model element names are limited. In summary, we restrict the *grammar of a natural language* in order to provide a formal basis for naming model elements (cf. Fig. 2).

Natural language grammars are usually defined by a formalism that consists of a *lexicon* and a *syntax specification* [27]. Such a grammar is complemented with naming conventions, which again consist of term and phrase structure conventions. Term conventions are specified by a *thesaurus* containing domain terms with a precise

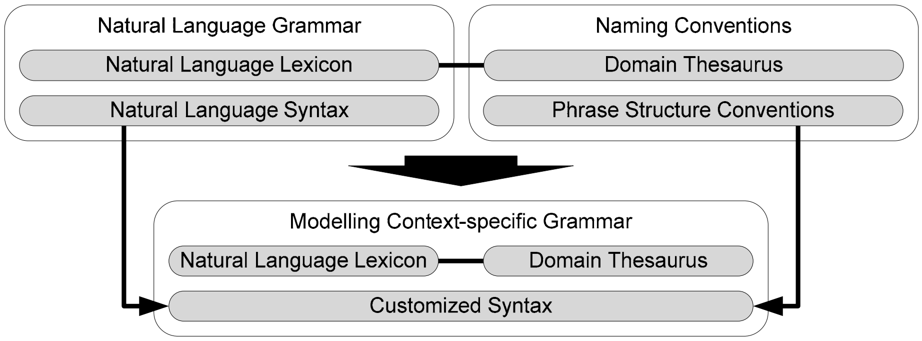


Fig. 2. Customizing the Natural Language Grammar with Naming Conventions

specification of their synonym, homonym, and word formation relationships as well as a textual description of their meaning. The thesaurus is then connected to the natural language's lexicon. Moreover, valid phrase structures are specified by *phrase structure conventions*. Hence, the natural language is customized for the needs of a specific modeling context. This allows for subsequent validation of the model element names and the enforcement of naming conventions. A conceptual overview of the naming conventions' specification is given in Section 3.2.

The thesaurus can be created from scratch, or by reusing possibly existing thesauri or glossaries. It includes single nouns, verbs, and adjectives that are interrelated. Other word classes are generally domain independent. Thus, as they are already included in the general lexicon, they do not need to be explicitly specified in the thesaurus. The terms in the thesaurus are linked to their synonyms, homonyms, and linguistic derivation(s) in the general lexicon. This additional term related information can be obtained from linguistic services, which already exist for different natural languages. For example, *WordNet* is such a lexicon service for the English language providing an online interface [21]. Therefore, in case of a later violation of the naming conventions by the modeler, synonymous or derived valid terms can be automatically identified and recommended. The terms specified are provided with short textual semantic descriptions, allowing modelers to look up the exact meaning of a term. The thesaurus should not be changed during a modeling project in order not to violate the consistency of application.

The naming conventions have to be specified once for every modeling context, whereas already existing conventions can be reused (in the following, cf. Fig. 3). Naming conventions are modeling language-specific. For example, functions in EPCs are labeled with activities (e.g., <*verb, imperative*><*noun, singular*>; in particular "check invoice") and events are labeled with states (for example <*noun, singular*><*verb, past participle*>; in particular "invoice checked") [24]. For each model element type at least one phrase structure convention has to be defined. For the sake of applicability, the conventions should be specified in a manner, which is compatible with the formalism of the natural language grammar.

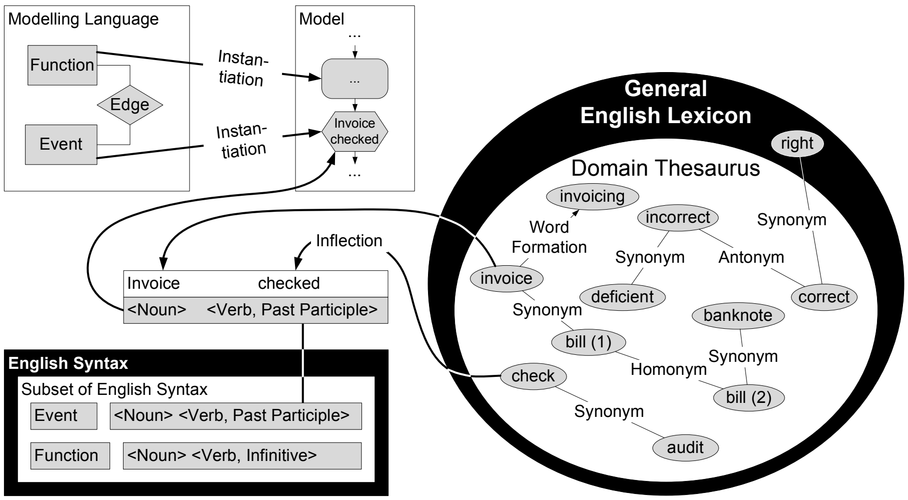


Fig. 3. Using Formalized Naming Conventions

The conventions should be defined by a project team consisting of domain experts and modeling experts. This means that the stakeholders responsible for the conventions should have thorough knowledge of the actual modeling context in order to reach a consensus. Most commonly, the thesaurus part of the conventions already exists in terms of corporate or domain-specific glossaries (e.g., [28, 29, 30]), which should be reused and adapted depending on the modeling situation (cf. Section 2.1).

During modeling, the entered model element names are verified simultaneously against the specified context-specific grammar. On the one hand, the structure of an entered model element name is validated against the customized syntax specification. On the other hand, it is checked whether the used terms are allowed. Nouns, verbs, and adjectives, meaning word classes covered by the thesaurus, are validated against it. Other word classes are validated against the natural language lexicon.

In case of a positive validation, the entered model element name is declared as valid against the modeling context-specific grammar. In case of a violation of one or both criteria, alternative valid phrase structures, terms or both are suggested based on the user input. The modelers themselves have to decide, which of the recommendations fits their particular needs. By looking up the semantic descriptions of the terms, modelers can choose the appropriate one. Alternatively, they can choose a valid structure as a pattern and fill in the gaps with valid terms on their own. However, it should be possible for the modeler to propose a new term with a short textual semantic description. In order not to distract the modeler from his current modeling session, the proposed term is accepted temporarily. In a next step, it is up to the modeling project expert team whether they accept the term or not. If the term is accepted, it is added to the thesaurus. Otherwise, the modeler is informed to revise the model element. Hereby, we ensure that equal model element names represent equal semantics, which is a precondition for comparability of conceptual models.

3.2 Conceptual Specification

In the following, we provide a conceptual framework for the specification and the enforcement of naming conventions using Entity-Relationship Models in (min,max)-notation [31] (cf. Fig. 4). *Phrase structure conventions (PSC)* are defined depending on distinct *element types* of conceptual modeling languages (e.g., *activities* in process models are named differently from *events*).

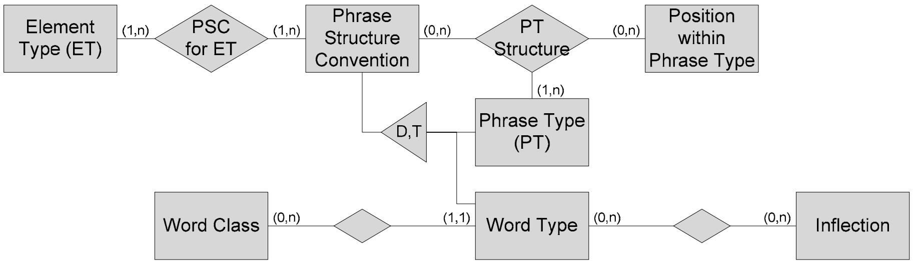


Fig. 4. Specification of Phrase Structure Conventions on Type Layer

Phrase structure conventions consist of *phrase types* or *word types*. A phrase type specifies the structure of a phrase, which can be used as a model element name. Therefore, we compose a phrase type recursively using further phrase types or word types. Representing atomic elements of a phrase type, word types are acting as placeholders for particular words. An example of a word type is <*noun, singular*>, an example of a phrase type is <*noun, singular*><*verb, infinitive*>. The composition of phrase types is specified by the *phrase type structure*. Here, we define the allocation of sub phrase types or word types to a phrase type and their *position* in the superordinate phrase type.

A word type consists of a distinct *word class* (noun, verb, adjective, adverb, article, pronoun, preposition, conjunction, or numeral) – and its *inflection*. Inflections can be specialized as case, number, tense, gender, mood, person, and comparative and these are usually combined. For example, a particular combined inflection is <*3rd person, singular*>. In respect to specific word classes, not every inflection is applicable. Based on the recursive composition of phrase types, it is possible to specify arbitrary phrase structure conventions.

Phrase structure conventions restrict the underlying English syntax and thus limit modelers in their freedom of naming model elements. In order to facilitate the synchronization between the syntax of the natural language and the applied phrase structure conventions, compatible formalisms for both syntax specifications are necessary. Hence, it should be possible to verify phrase structure conventions against the underlying natural language and to signalize potential conflicts directly during the specification process. For this purpose, we establish the connection to linguistic parsing approaches in Section 3.3.

Independently from their corresponding word class, particular uninflected words are called *lexemes* (e.g., the verb “check”). Inflected words are called *word forms* (e.g., past participle “checked” of the lexeme “check”). Word forms are assigned to the corresponding word types (i.e., their word classes and inflections) Thus, word forms represent lexemes of a particular word type (cf. Fig. 5).

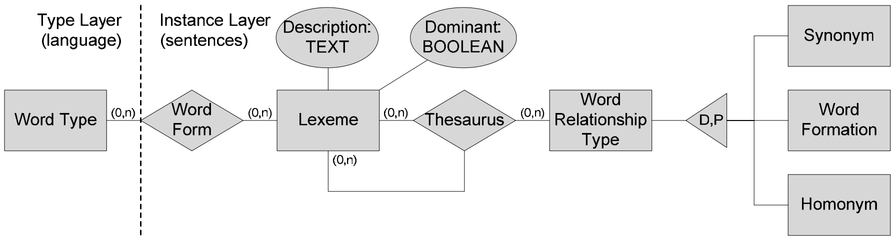


Fig. 5. Specification of Term Conventions on Instance Layer

In order to specify the domain *thesaurus*, we store the allowed words in the form of lexemes that are related by different *word relationship types*. They are specialized as *homonym*, *synonym*, and *word formation* relations. *Word formation* means that a lexeme originates from (an)other one(s) (e.g., the noun "control" originates from the verb "to control"). In case of synonym relations, one of the involved lexemes is marked as *dominant* to state that it is the valid one in the particular modeling context. Homonym relations are necessary in order to distinguish lexemes that consist of the same string but have a different meaning and to prevent errors during modeling. We use word formation relations to search for appropriate alternatives when a modeler has used invalid terms and phrase structures. For example, if the phrase "order clearance" violates the conventions, the alternative phrase "clear order" can be found via the word formation relation of "to clear" and "clearance". Based on the word relationship types, we connect the domain thesaurus to lexical services (cf. Section 3.1). To specify what is actually meant by a lexeme, a semantic *description* is added at least to each dominant lexeme. This way, modelers are enabled to check if the lexeme they have used actually fits the modeling issue.

3.3 Specification of Linguistic Restrictions

To assure the correctness of both specified phrase structure conventions and the structure and words of model element names, we make use of linguistic syntax parsing. According parsing methods detect the syntax of a given sentence based on so-called *universal grammar frameworks*. For example, an according parsing method analyzes the phrase "invoice checked" and returns the phrase type <*noun, singular*><*verb, past participle*> as well as the lexemes "invoice" and "check". For reasons of clarity, we do not introduce these methods in detail (cf. [27] for an overview). Given a phrase structure convention, according parsing methods are able to determine whether the convention complies with the syntax of a natural language. Furthermore, given a model element name, they determine whether the syntax of the name complies with the phrase structure conventions. This way we check the convention-related correctness of model element names concurrently.

In our approach, we parse sentences against the domain thesaurus and the restricted English syntax. If the terms used within model element names do not comply with the conventions, alternative but valid lexemes are searched in the domain thesaurus via the defined word relationships or in the general language lexicon and are proposed in the appropriate inflection form for proper use (cf. Fig. 6).

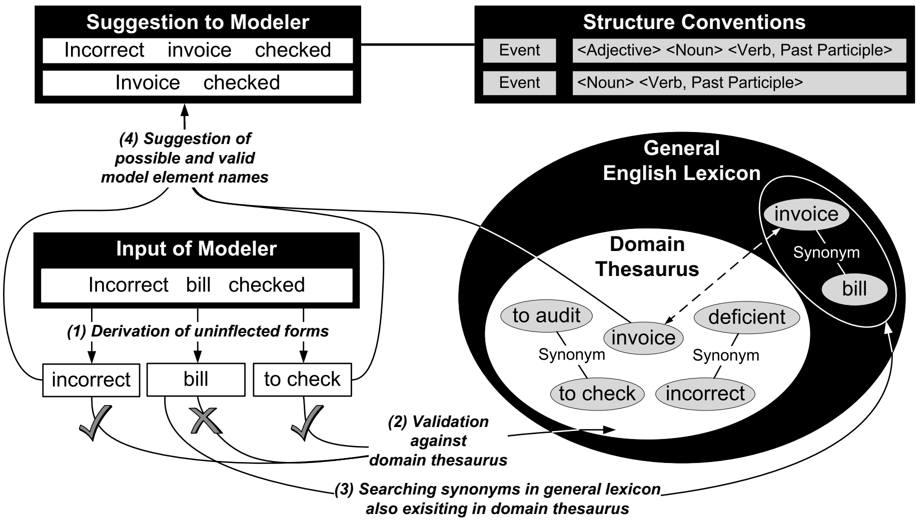

Suggestion to Modeler
Incorrect invoice checked
Invoice checked
Structure Conventions
Event
<Adjective> <Noun> <Verb, Past Participle>
Event
<Noun> <Verb, Past Participle>
(4) Suggestion of possible and valid model element names
General English Lexicon
invoice
Synonym
bill
Domain Thesaurus
Input of Modeler
Incorrect bill checked
(1) Derivation of uninflected forms
incorrect
bill
to check
to audit
invoice
deficient
Synonym
Synonym
to check
incorrect
(2) Validation against domain thesaurus
(3) Searching synonyms in general lexicon also exisiting in domain thesaurus

Fig. 6. Validation and Suggestion of Model Element Names

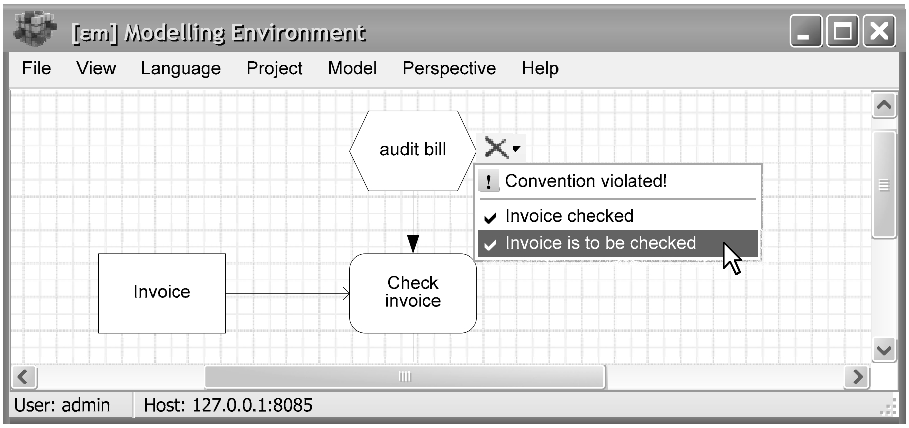
In particular, we decompose a model element name into single terms and derive their uninflected forms (1). In a next step, we validate the lexemes against the domain thesaurus (2). Lexemes contained in the domain thesaurus are denoted as valid. For those lexemes that do not exist in the domain thesaurus, we search synonyms in the general lexicon and match them against the domain thesaurus (3). If no such synonyms are available or a lexeme is not contained in the general lexicon, we exclude them from further validation steps. Based on the defined structure conventions, we suggest possible model element names to the modelers that contain the valid lexemes in the appropriate inflection form (4). If a phrase structure is violated in turn, alternative but valid phrase structures are proposed that contain the valid terms.

4 Modeling Tool Support

To validate the general applicability of our approach, we developed a modeling prototype. The way of navigating through the software and its handling is tightly connected to the procedure model motivated in Section 3.

As described above, the connection of our approach with modeling languages requires the adoption of the respective meta model. For the tool support, this indicates the necessity of meta modeling abilities, which is supported by our research prototype. Hence, virtually any modeling language that can be created or exists inside the prototype can be extended with naming conventions. The software follows a fat client/thin server three tier architecture, thus enabling distributed modeling. As the presentation layer is abstract, we chose the widespread drawing engine *Microsoft Visio*, accessing it with the *Microsoft .NET Framework*.

As a preliminary step, the person responsible for specifying the modeling conventions has to define the terms, which are allowed for the modeling context. Subsequently, the phrase structure conventions have to be specified. If the actual modeling context represents a domain which has been processed before, the existing set of terms and rules can be adapted to the current requirements. It is sufficient to add uninflected words, as the inflection can be looked up in the lexical services.



Convention violated!
Invoice checked
Invoice is to be checked

User: admin
Host: 127.0.0.1:8085

Fig. 7. Automatic Guidance in Order to Comply with Naming Conventions

In the next step, the user defines phrase structure conventions and connects them to those language elements for which they are valid. For example, it is necessary to create different phrase structure conventions for EPC events (i.e., separate conventions for *trigger events* and *result events*). Trigger events start functions and result events conclude them. Different phrase structures can be attached to each of them in regard to their different semantics. An example of a trigger event is "invoice is to be checked", hence an appropriate phrase structure convention called "Trigger" is <*noun, singular*><*verb, passive infinitive*>. With this phrase structure, a set of trigger events can be named.

However, different aspects might require additional phrase structures to be defined. For resulting events, an adequate phrase structure is <*noun, singular*><*verb, past participle*>, allowing phrases like "invoice checked". Once generated, the phrase structure conventions in combination with the domain thesaurus are used during modeling. Modelers get hints as soon as they violate a convention (cf. Fig. 7).

First, the modeler might have chosen invalid terms (e.g., *bill* instead of *invoice* or *audit* instead of *check*). As soon as (s)he has entered a phrase, it is parsed to determine its compliance with the conventions. The tool transforms every term into its uninflected form and compares it with the domain thesaurus. If the term is not found, synonymous valid terms are searched in the lexicon. If according alternatives are found, they are proposed to the modeler. Otherwise, (s)he has to rename the respective element – optionally by choosing a valid term from the domain thesaurus. Second, violations of phrase structure conventions are signaled and alternative valid structures are proposed.

Summarizing this example, the name *audit bill* is suggested to be changed to *invoice is to be checked*. Phrases corresponding with both the domain thesaurus and the phrase structure conventions are accepted without any feedback.

5 Conclusion and Outlook

Integrating naming conventions into conceptual modeling languages is promising for increasing the comparability of conceptual models. Two characteristics are significant to avoid common problems:

- Defining and providing naming conventions *previously to modeling* is the basis for *avoiding* naming conflicts rather than resolving them. Therefore, time-consuming alignment of namings becomes dispensable.
- *Guiding the modeler automatically during modeling* is of substantial importance, since only this way the compliance with the modeling conventions can be assured.

Certainly, specifying naming conventions in the proposed way is time-consuming. Our approach is therefore mainly suited for large-scaled regionally distributed modeling projects. Nevertheless, for every project, business domain, or company, the conventions have to be specified only once and are reusable. Moreover, term models, thesauri and/or glossaries that may already exist in companies or business domains can be reused. Furthermore, our approach is restricted to models that are developed from scratch. It is not suitable for existing models that are to be made comparable, as it can be seen by the ontology-based approaches presented in Section 2.

Future research will focus on further evaluating the proposed approach. In the short-term, we will instantiate the approach for different modeling languages, different natural languages and different application scenarios. In particular, we will evaluate the capability of our approach to increase the efficiency of distributed conceptual modeling and its acceptance. To assure the applicability of the approach, we will enhance the demonstrator software in order to make it usable in practice.

Moreover, as linguistic grammar approaches and according parsers usually vary in terms of linguistic coverage, it has to be empirically evaluated which ones provide the best coverage for certain natural languages and application scenarios. In addition to conducting this in batch using artificially prepared test samples, we will implement a set of different parsers in our tool and conduct empirical evaluations of these in real-life modeling settings. In the course of evaluation, we will also investigate whether ambiguities play a role in model element names. For example, the sentence “They hit the man with a cane” is ambiguous, even if the meanings of all of the used words are considered definite. Thus, we will perform further studies on existing conceptual models and determine if phrase structures promoting ambiguities are common in conceptual modeling. A result of this analysis could be a recommendation to restrict phrase structure conventions to phrases that do not lead to ambiguities.

References

1. Batini, C., Lenzerini, M., Navathe, S.B.: A Comparative Analysis of Methodologies for Database Schema Integration. ACM Computing Surveys 18(4), 323–364 (1986)
2. Lawrence, R., Barker, K.: Integrating Relational Database Schemas using a Standardized Dictionary. In: Proceedings of the 2001 ACM symposium on Applied computing (SAC), Las Vegas (2001)

3. Hadar, I., Soffer, P.: Variations in conceptual modeling: classification and ontological analysis. Journal of the AIS 7(8), 568–592 (2006)
4. Phalp, K., Shepperd, M.: Quantitative analysis of static models of processes. Journal of Systems and Software 52(2-3), 105–112 (2000)
5. Vergidis, K., Tiwari, A., Majeed, B.: Business process analysis and optimization: beyond reengineering. IEEE Transactions on Systems, Man, and Cybernetics 38(1), 69–82 (2008)
6. Batini, C., Lenzerini, M.: A Methodology for Data Schema Integration in the Entity Relationship Model. IEEE Transactions on Software Engineering 10(6), 650–663 (1984)
7. Bhargava, H.K., Kimbrough, S.O., Krishnan, R.: Unique Name Violations, a Problem for Model Integration or You Say Tomato, I Say Tomahto. ORSA Journal on Computing 3(2), 107–120 (1991)
8. Chen, P.P.-S.: The Entity-Relationship Model: Toward a Unified View of Data. ACM Transactions on Database Systems 1(1), 9–36 (1976)
9. Gruber, T.R.: A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition 5(2), 199–220 (1993)
10. Guarino, N.: Formal Ontology and Information Systems. In: Guarino, N. (ed.) Proceedings of the 1st International Conference on Formal Ontologies in Information Systems, Trento, pp. 3–15 (1998)
11. Greco, G., Guzzo, A., Pontieri, L., Saccá, D.: An ontology-driven process modeling framework. In: Galindo, F., Takizawa, M., Traunmüller, R. (eds.) DEXA 2004. LNCS, vol. 3180, pp. 13–23. Springer, Heidelberg (2004)
12. Born, M., Dörr, F., Weber, I.: User-friendly semantic annotation in business process modeling. In: Weske, M., Hacid, M.-S., Godart, C. (eds.) Proceedings of the International Workshop on Human-Friendly Service Description, Discovery and Matchmaking (Hf-SDDM 2007). 8th International Conference on Web Information Systems Engineering (WISE 2007), Nancy, pp. 260–271 (2007)
13. White, S.A., Miers, D.: BPMN Modeling and Reference Guide. Understanding and Using BPMN. Lighthouse Point (2008)
14. Höfferer, P.: Achieving business process model interoperability using metamodels and ontologies. In: Österle, H., Schelp, J., Winter, R. (eds.) Proceedings of the 15th European Conference on Information Systems (ECIS 2007), St. Gallen, pp. 1620–1631 (2007)
15. Ehrig, M., Koschmider, A., Oberweis, A.: Measuring Similarity between Semantic Business Process Models. In: Proceedings of the 4th Asia-Pacific Conference on Conceptual Modelling (APCCM) 2007, Ballarat (2007)
16. Koschmider, A., Oberweis, A.: Ontology Based Business Process Description. In: Enterprise Modelling and Ontologies for Interoperability, Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability, Co-located with CAiSE 2005 Conference, Porto (2005)
17. Sabetzadeh, M., Nejati, S., Easterbrook, S., Chechik, M.: A Relationship-Driven Framework for Model Merging. In: Proceedings of the Workshop on Modeling in Software Engineering. 29th International Conference on Software Engineering, Minneapolis (2007)
18. Rosemann, M.: Complexity Management in Process Models. Language-specific Modelling Guidelines. Komplexitätsmanagement in Prozeßmodellen. Methodenspezifische Gestaltungsempfehlungen für die Informationsmodellierung (in German). Wiesbaden (1996)
19. Kugeler, M.: Organisational Design with Conceptual Models. Modelling Conventions and Reference Process Model for Business Process Reengineering Informa-tionsmodellbasierte Organisationsgestaltung. Modellierungskonventionen und Referenzvorgehensmodell zur prozessorientierten Reorganisation (in German). Berlin (2000)

20. Rosemann, M.: Preparation of Process Modeling. In: Becker, J., Kugeler, M., Rosemann, M. (eds.) Process Management – A Guide for the Design of Business Processes, Berlin, pp. 41–78 (2003)
21. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. Cambridge (1998)
22. Rizopoulos, N., Mçbrien, P.: A General Approach to the Generation of Conceptual Model Transformations. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 326–341. Springer, Heidelberg (2005)
23. Bögl, A., Kobler, M., Schrefl, M.: Knowledge Acquisition from EPC Models for Extraction of Process Patterns in Engineering Domains. In: Proceedings of the Multi-Conference on Information Systems Multikonferenz Wirtschaftsinformatik 2008 (MKWI 2008)(in German), Munich (2008)
24. Scheer, A.-W.: ARIS – Business Process Modelling, 3rd edn., Berlin (2000)
25. Schmid, H.: Probabilistic Part-of-Speech Tagging Using Decision Trees. In: Proceedings of the First International Conference on New Methods in Natural Language Processing, Manchester, pp. 44–49 (1994)
26. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly 28(1), 75–105 (2004)
27. Kaplan, R.M.: Syntax. In: Mitkov, R. (ed.) The Oxford handbook of computational linguistics, Oxford, pp. 70–90 (2003)
28. Automotive Thesaurus, `http://automotivethesaurus.com`
29. Tradeport – Reference Library for Global Trade, `http://tradeport.org/library`
30. WWW Virtual Library: Logistics, `http://logisticsworld.com/logistics/glossary.htm`
31. ISO: ISO/TC97/SC5/WG3: Concepts and Terminology for the Conceptual Schema and the Information Base (1982)