

Conflict Resolution in the Collaborative Design of Terminological Knowledge Bases

Gilles Falquet, Claire-Lise Mottaz Jiang

Centre universitaire d'informatique, University of Geneva¹
24, rue du Général-Dufour, CH-1211 Genève 4, Switzerland
Gilles.Falquet@cui.unige.ch Claire-Lise.Mottaz@eti.unige.ch

Abstract

Designing a terminological knowledge base consists in collecting terms and associating them to their definition. Our objective is to define a process model to support this design task in a collaborative work environment. The proposed concept model is based on terminological logic and the issue-based model IBIS. The terminological logic part is intended to formally express definitions and associate them to terms and points of view. The process model we define is based on a cyclic conflict resolution process. It includes a formal concept comparison operation, to highlight definition conflicts and their nature, and other operations (derivation, conjunction, disjunction, etc.) to solve the detected conflicts. The IBIS part of the model enable users to express and record issues, positions, arguments and endorsements that occur during conflict resolution.

1 Introduction

1.1 Background

Terminology is about identifying, describing and naming a field's concepts. Terminology's basic elements are: concepts, terms, definitions and fields. A concept is described by a definition and is named by a term. As a rule, a term can only refer to a single concept within a field. The elaboration of terminological dictionaries and concept bases is generally intended to make translators' job easier or to ensure a better communication between a field's specialists. In the recent years it has become obvious that this terminological work is crucial in information systems design and particularly in knowledge management.

Everyone has his own perception of real world's objects. Thus, when a group of people is building up a concept base or an information system, its members often don't agree on the meaning of the terms, i.e. there are vocabulary conflicts. Surprisingly, although there are many types of concept bases, none of them allows, as far as we know, to store and manage multiple, not necessarily coherent, points of view for a concept's definition. As a result, the choice of a definition or a term must usually be done before it can be inserted into the concept base. So we can say that concept models only allow to store the conceptualization's result but don't directly support the conceptualization process.

1.2 Related Work

Traditional terminology banks, such as Eurodicautom (European Union), Termium (Canada), Lingua-PC (Switzerland, Canton of Bern) or BD-TERM (University of Geneva) [Deb98], [Pul88] represent a first type of concept bases. Concepts are described using textual definitions and other terminological descriptors (synonym, context, source, note). In these terminology banks it could be possible (even if it is not usually done) to store multiple points of view, for instance several definitions for a concept, because the record associated with each term is typi-

1. This work is a part of a joint project between the CUI and the ETI (School of Translation and Interpretation) at the University of Geneva

cally stored as formatted text. But as concept representations are not formalized, it is difficult to apply automatic processing on them.

In terminological knowledge representation systems, (KL-ONE [Bra91], ALCNR [Buc93], etc.) concepts are characterized by a set of roles which link them to other concepts in the base. In this case, definitions are not textual but formalized, thus allowing some automatic processing. Nevertheless, in this case we have to face the opposite problem: it is not possible, with this kind of formalism, to handle several definitions for a single concept.

The ConcepTerm model [Ber94], [Sin95] is relatively close to classic terminological knowledge representation systems. Concepts are defined by a set of pairs <characteristic; value>. The goal of ConcepTerm was to enable the search for equivalent terms in different languages by comparing related concepts' definitions. This can give interesting suggestions on how to compare concepts, but this model does not allow to store several definitions for a concept.

The Co4 system [Euz96] suggests an interesting approach for the collaborative building of a consensual knowledge base from several individual bases. The bases are organized in a tree in which leaves are the individual bases and each node represents the consensual base of the subtree. The tree's root is the global consensual base. With Co4, the rule is: before inserting a piece of knowledge into a consensual base, one must be sure that all the bases of the subtree agree with it. Co4 is a kind of multi point of view system: knowledge in a consensual base is not the same as knowledge in individual bases. It is however difficult to have a global view, since the different points of view are dispersed in several bases.

Collaboratively designing and building a concept base can also be seen as a decision making process: for each concept and each point of view it is necessary to choose one definition among those which are suggested by the group members. There exist several models for decision making support in an argumentative environment, such as IBIS [Con89], [Con96], [Gro], [Kun72] QOC and DRL [Buc97], [Stu98]. This kind of models will give us a basis for the creation of a multi point of view concept model.

When several points of view are available, it could help to have tools for comparing and manipulating them. So, as we are mainly interested in managing multiple points of view for concepts definitions, we have to quote the works of Shaw and Gaines on conceptual systems comparisons [Sha89]. Since the method of Gaines and Shaw aims at comparing two or more different conceptual systems, it takes into account object names, attributes and values. For instance, it can compare attributes values even if the attributes names do not match.

The following table explains some of the terms that we will use later. It is taken from [Sha89] and indicates the possible situations resulting from the comparison of two or more conceptual systems.

		Terms	
		Same	Different
Concepts	Same	<p>Consensus</p> <p>People use the same terms to name the same concepts</p>	<p>Correspondence</p> <p>People use different terms to name the same concepts</p>
	Different	<p>Conflict</p> <p>People use the same terms to name different concepts</p>	<p>Contrast</p> <p>People use different terms to name different concepts</p>

One can remark that Shaw and Gaines' method is meant to compare two or more different conceptual systems, whereas our main preoccupation is what to do with one incoherent system,

build collaboratively. Their method will nevertheless give us suggestions on how to define our concepts comparison operation. These remarks are also applicable to the method presented by Dieng [Die97] for modeling knowledge of multiple experts. (This method is based on the comparison of conceptual graphs.) It is also worth noticing that using different terminologies doesn't inevitably imply a contrast: maybe people just have a different level of abstraction.

1.3 Multiple points-of-view

The KRL, LOOPS, ROME, VIEWS and TROPES [Mar93] models propose different kinds of solutions for the management of multiple points of view. However, these models all rely on the hypothesis that points of view are partial representations of a unique coherent set of objects. We focus on another situation: when building the concept base, each person (or group of people) has his own incomplete perception of the field; the sum of all individual perception giving an incoherent representation of that field. This difference between basic hypothesis stems from the fact that the model we are presenting in this paper is meant to support group knowledge acquisition and building whereas the others are more adapted to a collective use of already build knowledge.

We consider point of view as a mean to solve definition conflicts. Namely, when two definitions are proposed for the same term, the multi point of view approach allows to keep both definitions, provided they belong to different points of view. For example, it would be easy to accept that a cashier and a mathematician do not define the concept of addition in the same way. Since we do not consider points of view as partial representation of a unique definition, we can even accept definitions which are not completely compatible. This is to reflect the fact that there is generally no strict border to the extension of a concept. For instance, where is the border between red objects and brown objects? Nevertheless, it is clear that the definition must not be contradictory. In addition, points of view are not intended to hide the conflicts and to please each participant, they must in fact correspond to a real application (e.g. sales, engineering, accounting) or group of users of the knowledge base.

1.4 Organization of this paper

The rest of this paper is organized as follows. Section 2 presents the ConceptIBIS model. Section 3 introduces the concept comparison and derivation operations which will be used in the conflict resolution process. Section 4 presents the conflict resolution process. And finally, section 5 gives a conclusion.

2 The ConceptIBIS model

When building a terminological concept base, two essential yet reciprocal problems occur: How to define the concept corresponding to a term? What term to use to name a concept with this or that definition? When a group of people is building a terminological concept base, it can lead to several situation corresponding to these two types of problems. Specifically, there can be:

- several different definitions for a single term
- several different terms for a single definition

The main goal of the ConceptIBIS model is to provide a background for 1) highlighting the above-mentioned situations and 2) solving these situations in a multi point of view context. The resolution of a definition conflict can lead to several situations: the two definitions are accepted and each one is linked to a different point of view; or one tries to create a single definition from the two conflicting ones; or one accepts that there are in fact two different concepts (for instance if the definitions are contradictory).

2.1 Structure of the model

ConceptIBIS is based on ConcepTerm. An argumentative part based on IBIS has been added to enable the management of multiple points of view. The purely terminological part of the model consists of *concepts*, *terms*, *definitions*, *fields*, and *points of view*. A concept definition comprises a set of characteristics with their respective values, the structure of a definition will be detailed in the next section.

Since a concept is an abstraction, a mental representation of real object, it doesn't have a material existence. Thus it must always be associated to either a term of a definition that represent it. This fact is represented in the model by associations between the classes *Concept* and *Term*, and *Concept* and *Definition*. In order to implement the multiple point-of-view approach, each definition must be attached to at least one point of view on the concept's field. Furthermore, two definition may be associated with the same concept only if they belong to different points of view. Violation of this rule means that there is a definition conflict.

In IBIS, there are three types of elements: issues, positions, and arguments. A position can be seen as a way to solve a given issue, and an argument may be in favor or against a position. In ConceptIBIS, we use the IBIS model to formalize and keep track of the conflict resolution process. Definition conflicts are the issues; a position corresponds to the choice of an operation in the conflict resolution process (defined in section 4); and arguments are in favor or against choices. Each operation is related to its operands which are objects of the model (definitions, points of view, concepts, etc.). For instance, the operands of an operation "associate definition *d* with point of view *v*" has two operands of type *Definition* and *Point of view* respectively. Since the concept base construction process involves modifying definitions, it is necessary to keep all the versions of a definition which have been involved in a conflict resolution operation. Thus each definition version is linked to the previous version. Finally, an endorsement is a recognition by some authority that a given definition - concept - term association is valid. Figure 1 shows a formal definition of the structure of ConceptIBIS (using a UML-like notation)

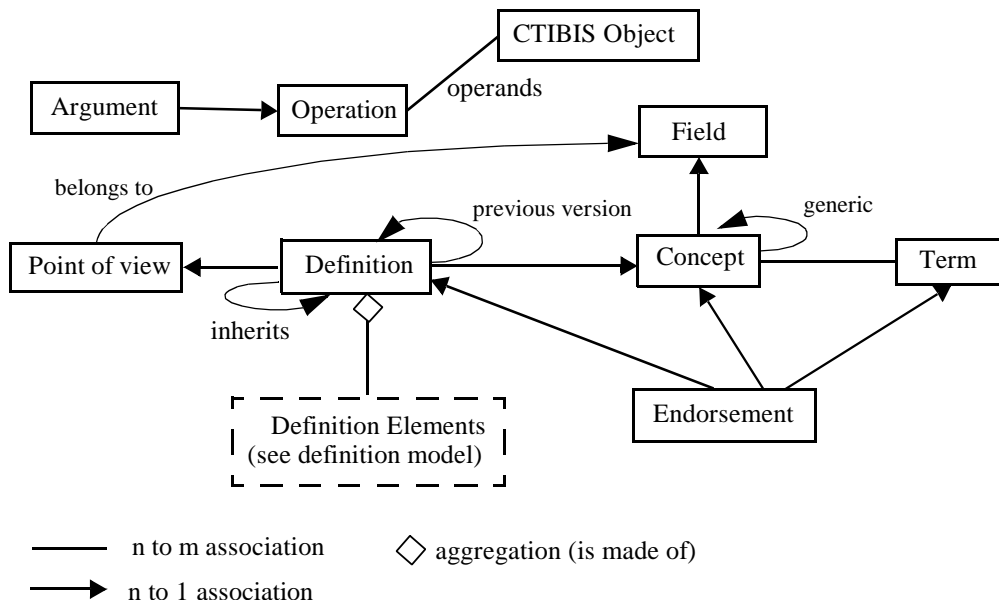


Figure 1 The ConceptIBIS model in UML

The *generic* association between definitions is a syntactic relationship which means that a definition inherits definition elements from another one. The *generic* association between con-

cepts is semantic one, meaning that the generic concept has a wider interpretation (set of instances). The notion of synonym is implemented by connecting two terms to the same concept.

2.2 Definition Model

We use the model which was developed for the creation of multilingual concept bases in the ConcepTerm project [Ber94]. The model we use here is a slight extension of the model presented in [FM99]. The extension consists in introducing number constraints as a separate construct instead of using “number” characteristics¹.

A definition is a specialization of a more general definition: it is composed of a set of characteristics. A characteristic has a name, a quantifier or a number restriction and a value definitions. A value definition is itself a definition, it specifies which object categories are allowed for a given characteristic. Formally, a concept definition is a statement which follows the following syntax:

ConceptDefinition ::=

definition DefinitionId **generic** DefinitionId **characteristics** Characteristic*

Characteristic ::= [**all** | NumberRestriction] CharacteristicName ":" Value

NumberRestriction ::= "<" PositiveNumber "," NonNullPositiveNumber ">"

PositiveNumber ::= "0" | "1" | "2" | ... | "*"

NonNullPositiveNumber ::= "1" | "2" | ... | "*"

Value ::= [**not**] Term | Disjunction | Conjunction | Characteristic

Disjunction ::= "{" Value* "}"

Conjunction ::= "(" Value* ")"

Where * denotes 0, 1 or several occurrences of an element; [] denotes 0 or 1 occurrences and | denotes alternative.

Example. A definition for the concept [wardrobe]²

definition wardrobe

generic storage_furniture

characteristics

Dimension : big,

Part : (type : door)

Part : <2, *> (type : shelf)

Part : (type : body)

all Main_Use : (verb : store, object : {linen ; clothes})

Terms which appear in a definition indicate predefined concepts, i.e. concepts for which there is not explicit definition in the concept base (atomic concepts). The atomicity of a concept is not an absolute notion, it is relative to a field. For instance, *wood* can be regarded as atomic within the furniture field whereas it will be explicitly defined when talking about building materials.

1. This extension was dictated by early results we obtained with the comparison algorithm. It is intended to reduce the relative importance of having equalities on number restrictions when comparing concepts.

2. from the “Furniture” concept base of ConcepTerm project

It is sometimes useful to view a concept definition as a syntax tree with each arc representing a characteristic. In particular, we will define the definition comparison operation in terms of tree transformation. The following figure shows the tree representation of the previous example (wardrobe).

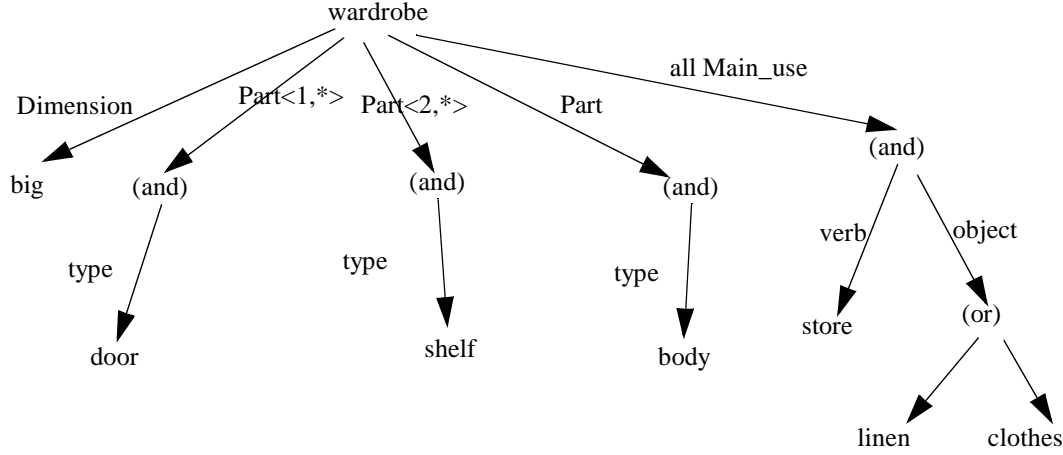


Figure 2 A definition in tree form

The semantics of a definition is a subset of an interpretation domain. This subset corresponds to the extension a concept. A knowledge base is a set of definitions. An interpretation I of a knowledge base (KB) is composed of

- a set Δ^I (the interpretation domain)
- for each elementary concept e (designated by a term), an interpretation $e^I \subseteq \Delta^I$
- for each characteristic R , a relation $R^I \subseteq \Delta^I \times \Delta^I$

The interpretation of a definition is obtained by applying the following rules :

$$I(\text{generic } G \text{ characteristics } K_1 K_2, \dots K_n) = I(G) \cap I(K_2) \cap \dots \cap I(K_n)$$

$$I(R: \langle \min, \max \rangle V) = \{ o \mid \min \leq \text{card}\{p \in I(V) \mid (o, p) \in R^I\} \leq \max \}$$

$$I(\text{all } R: V) = \{ o \mid \forall p. (o, p) \in R^I \Rightarrow p \in I(V) \}$$

$$I((C_1, C_2, \dots, C_n)) = I(C_1) \cap I(C_2) \cap \dots \cap I(C_n)$$

$$I(\{C_1, C_2, \dots, C_n\}) = I(C_1) \cup I(C_2) \cup \dots \cup I(C_n)$$

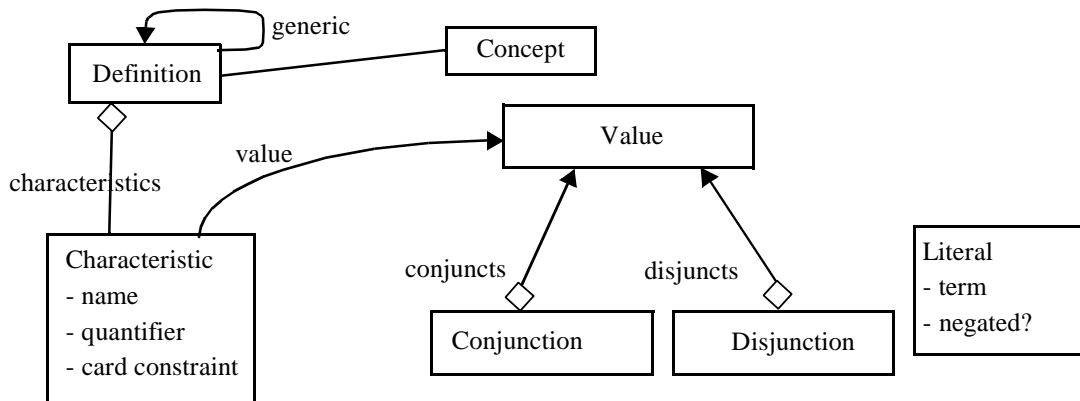
$$I(\text{term}) = \text{term}^I,$$

$$I(\text{not term}) = \Delta^I \setminus \text{term}^I,$$

Commutativity and associativity of the union and the intersection imply that the order in which the elements of a concept definition appear has no importance (interpretation remains unchanged under element permutation).

This model and its semantics are close to the terminological knowledge representation model *ALCNR* [Buc93]. The main difference lies in the number restriction construct. In *ALCNR* a number restriction applies to a role ($\leq n \text{ Role}$ and $\geq n \text{ Role}$) while in *ConceptTerm* it applies to a role and a value (*Role: $\langle \min, \max \rangle \text{Value}$*), meaning that an instance of this concept must be linked to at least *min* and at most *max* instances of *Value* through *Role*. If there are several characteristics with the same name, an equivalent *ALCNR* definition can be obtained by introducing new role names.

In terms of concept base, a definition is represented by objects of the classes and associations shown on Figure 3. In the ConceptIBIS system, definitions are actually stored under this form.



Characteristic, Conjunction, Disjunction, and Literal are subclasses of Value

Figure 3 Structure of a definition in UML-like notation

3 Operations for collaborative work

The main operations of the ConceptIBIS model are the definition comparison (to detect conflicts), and definition derivations (to help the resolution process.)

3.1 Definitions comparison

Comparison is the basic operation to identify consensus and divergence, identify synonyms, etc. It is central in a process of collaborative building of concept bases. The comparison of two definitions is done by comparing their respective sets of characteristics. For this operation to be useful, it must indicate precisely the differences that exist between two definitions. A boolean comparison is not enough (A is equal to B or A is different from B); neither is a comparison that calculates a distance between two concepts and only gives a positive real number (whatever the sophistication of the calculation). One should also note that the n-dimensional distance is not applicable since characteristics may be multivaluated.

Our approach, which is mostly syntactic consists in expressing the difference between two definitions C1 and C2 as modifications. A *modification* of a definitions C1, regarded as a tree, is a labeled tree which is an extension of C1. Arcs coming from C1 may remain unlabeled (unchanged) or be labeled with [-] to indicate subtree removal. Added subtrees are labeled with [+] on their to level arcs. Similarly, number constraints may be added and removed. A difference between definitions C1 and C2 is a modification of C1 which, when evaluated, yields C2, and has minimal complexity.

Example. Let C1 and C2 be the definitions shown on Figure 4. The following expression is a modification of C1 that yields C2:

- [+] Dimension: big
- Part: [-]<1,*>[+]<0,*>(type: door, [+] material: (type: pane)),
- Part: [-]<1,*>[+]<2,*>(type: shelf),
- [-] Part: <1,*>(type: body)
- Main_Use: (verb: store, object: {[-] linen; [-] clothes; [+] books})

<p>C1 = generic storage_furniture characteristics Dimension: big Part: <1,*> (type: door), Part: <1,*> (type: shelf), Part: <1,*> (type: body) Main_Use : <1,1> (verb: store, object: {linen; clothes})</p>	<p>C2 = generic storage_furniture characteristics Part: <0,*>(type: door, material: (type: pane)), Part: <2,*>(type: shelf) Main_Use: <1,1> (verb: store, object: books)</p>
--	---

Figure 4 Two concept definitions

The complexity of a modification depends on the number of modification labels it has and the depth at which these labels occur. A modification label at level n has weight $1/p^n$, where p is an integer parameter greater than 2.

If a modification is composed of characteristics K_1, \dots, K_n , its complexity $\chi(M)$ is defined as the mean of the characteristics' complexities:

$$\chi(M) = (\chi(K_1) + \dots + \chi(K_n))/n$$

The complexity of a labeled characteristic is recursively defined by the following rules:

adding/removing a characteristic

$$\chi([\pm] R \langle min, max \rangle : V) = \chi([\pm] \text{ all } R : V) = 1$$

modification of a number constraint

$$\begin{aligned} \chi(R : [-] \langle min, max \rangle [+] \langle min', max' \rangle V) \\ = b_{comp} + 1/p \chi(V) \text{ if } \langle min, max \rangle \text{ et } \langle min', max' \rangle \text{ are compatible} \\ = b_{incomp} + 1/p \chi(V) \text{ if } \langle min, max \rangle \text{ et } \langle min', max' \rangle \text{ are incompatible} \end{aligned}$$

(Constraints are incompatible if the set of integer they define have an empty intersection, b_{comp} and b_{incomp} are real number parameters satisfying $b_{comp} < b_{incomp}$ and $b_{incomp} + 1/p \leq 1$.)

modification of the values (values may be characteristics, conjunctions, disjunctions, or literals)

$$\chi(R \langle min, max \rangle : V) = \chi(\text{ all } R : V) = 1/p \chi(V)$$

adding or removing a conjunction, a disjunction, or a literal

$$\chi([\pm](V_1, \dots, V_n)) = \chi([\pm]\{V_1; \dots; V_n\}) = \chi([\pm] \text{ not } [-] \text{ term } [+] \text{ term}') = 1$$

adding or removing an element within a conjunction or a disjunction

$$\begin{aligned} \chi((V_1, \dots, V_n)) &= \chi(V_1) + \dots + \chi(V_n)/n \\ \chi(\{V_1; \dots; V_n\}) &= \chi(V_1) + \dots + \chi(V_n)/n \end{aligned}$$

Formal definitions of the notions of modification and difference, as well as a discussion on the computational and semantic properties of the difference can be found in [FM99].

It is important to note that this notion of difference is essentially syntactic. However, when the complexity of a difference is null, we are sure that both definitions have the same interpretation, but the converse is not true. In fact, computing a semantic distance would require to know the interpretation of each predefined term, which is not the case in the bases we consider. As mentioned before, what is most important for conflict resolution is to have a clear view of what makes two definitions different. In addition, we are interested in finding syntactic differences even if they have no semantic effect. This is typically what happens when two designers have used different characteristic names to mean the same thing. Since we consider this situation as a conflict, it must be detected when computing differences.

The complexity of the distance and difference calculation is exponential, because in all cases of (and), (or) and multivaluated characteristics (several characteristics with the same name), one needs to try all possible permutations to find which one minimizes complexity. However, in the real cases that we met, the size of the permutations was limited.

3.2 Manipulation operations: derivation

Once comparison has been carried out, one needs a few manipulation operations in order to make further steps towards consensus. Basically, manipulation operations should enable the modification of existing definitions. But as endorsements refer to terms and definitions, modifying a definition could invalidate an endorsement. Similarly, conflict resolution arguments refer to operations and operands and could be invalidated by definition changes. To avoid this situation, it is forbidden to change definitions that are referenced from an endorsement or argument. Every operation must be done either on a new version of an existing definition or on a completely new definition (both are basically a copy of the original definition).

In other words, one can say that all manipulation operations are grouped under the “derivation” label. A *derivation* is a new definition which is created from an existing definition by either

- modifying the name and/or the value of one or more of its characteristics, or
- adding one or several new characteristics, or
- removing one or several characteristics.

A derivation can either be considered as a new version of the original definition or as a completely new definition. (A new version of a definition still refers to the same concept, whereas a new definition corresponds to a new concept.). The following two operations are intended to automatically produce derivations that can help in the resolution process.

Definition intersection

The intersection of two definitions A and B is a new definition that possesses only their common parts. This operation depends on the difference between A and B that is chosen. If D is a difference (labeled tree) from A to B, the intersection corresponding to D is obtained by removing all the [-] or [+] labeled subtrees that belong to a conjunction (including the top-level characteristics); retaining all the subtrees that belong to a disjunction; and removing all the [-] or [+] labeled cardinality constraints and universal quantifiers. One can see that the intersection creates a definition that is more general than the intersected definitions (i.e. its interpretation will always contain the interpretation of each intersected definition).

Definition union

The union is the dual of the intersection operation. It retains all the characteristics of both definitions which are in a conjunction and retains only the common characteristics in disjunctions.

It creates a definition whose interpretation is included in each one of the original definition interpretation.

Although these two operations do not automatically solve definition conflicts, they produce different alternatives that can be examined by the designers. This corresponds to the well known conflict resolution technique which consists in generating and proposing new alternatives.

4 Conflict analysis and resolution

In ConceptIBIS, we use the term “conflict” when:

- two (or more) terms designate the same concept,
- in a given field, two (or more) definitions describe the same concept and they belong to the same point of view.

The first type of conflict can be solved by answering to the question: “Are those two terms synonyms?”. In the case of a positive answer, a synonymy link is created between them. Otherwise, it is necessary either to remove one term, or to create a new concept for one of these terms. Solving definition conflicts will be the main topic of this section. We will first situate the conflict resolution task within the terminological knowledge base building process. Then we will show what can be done automatically to analyze definition conflicts and indicate which operations can be used to resolve them.

4.1 The collective creation process

The collaborative building of a terminological concept base with ConceptIBIS is an iterative process. We can see three main phases:

- 1) Free creation of terms, definitions and concepts.
- 2) Deliberations: participant can show their agreement or disagreement with the definitions by creating positive or negative endorsements.
- 3) Conflict analysis and resolution

The conflict analysis and resolution phase can be applied locally to a part of the knowledge base, while the other parts remain in phase 1 and 2. Moreover, it is not compulsory to resolve all the conflicts to return to phase one. The idea is that the knowledge base is built progressively and also becomes gradually more consistent.

4.2 Using comparisons to analyze conflicts

Testing generalizations and specializations

If there exist a path F from D1 to D2 which only contains [+] in conjunctions and [-] in disjunctions and which only restricts cardinality constraints and adds universal quantifiers, then it is sure that the interpretation of D1 will contain the interpretation of D2 (this is true because there are no negations outside literals and also because the $\langle 0,0 \rangle$ cardinality constraint is forbidden and replaced by a universal quantifier). One will then say that F proves that D2 is a specialization of D1. If D2 is a specialization of D1.

Testing compatibility

D1 and D2 are incompatible if no object can fulfil both definitions at the same time. With the analysis of differences between D1 and D2 it is possible to prove some cases of incompatibility. However, one can not prove every incompatibility case because it would require a precise

knowledge of concepts corresponding to terms used in literals. One can enumerate a set of inference rules which allow to discover some incompatibilities between definitions. (incoherent concept detection in CLASSIC [Bra91])

A difference F between two concepts definitions or between two value definitions proves the incompatibility of two definitions in the following cases:

Replacement of a literal by its negation,

F contains: [+] term [-] not term or [-] term [+] not term

Replacement by an incompatible cardinality constraint

F contains: R [-] <min, max>[+]<min', max'>: D
where <min, max> and <min', max'> are incompatible and D only contains operations corresponding to a generalization or a specialization.

Incompatibility between an existential characteristic and an universal characteristic

If one can prove the incompatibility between D and D' (by analysing their differences) and F contains: [+] all R : D and [-] R : D' (or the opposite).

Conjunction

if one can prove that D_i and D_j are incompatible and F contains: ($D_1, \dots, [+] D_i, \dots, [-] D_j, \dots, D_n$).

Disjunction

if for $1 \leq i \leq n$ and $1 \leq j \leq m$ one can prove that D_i is incompatible with E_j and F contains : $\{[+]D_1; \dots; [+]D_n; [-]E_1; \dots; [-]E_m\}$.

Compatibility is independent from distance (the complexity of a difference) between definitions. For example, if two definitions have no common characteristic, they will be perfectly compatible even if the distance is large.

The compatibility of two definitions does not necessarily imply that they represent the same concept.

A human intervention is required to complete the diagnostic, that is to identify semantic incompatibilities that difference analysis can not detect.

Compatibility should be regarded as a constraint rule that the knowledge base must validate in order to be in a coherent state. For the knowledge base to be in a coherent state, two definitions of a same concept, must be compatible and belong to different points of view. However, during the development of the base, incompatibilities are allowed.

4.3 The resolution process

The concept base is in a *coherent* state if, for each concept, there is at most one current definition per point of view, and if all these definitions are compatible. When a definition conflict occurs, there are three possibilities to solve it:

- Consensus: only one definition is kept. For that purpose, one can either remove one of them or merge the two basic definitions, with the union operation for example.
- Contrast: one decides that the two definitions correspond to two different concepts. One can then either create a new concept and a new term for one of the definitions, or create a new concept, keep the same term and link to another field.
- Different points of view: the two definitions are kept but each one is linked to a different point of view.

Remark: We use the terms: “conflict”, “consensus”, “correspondence” and “contrast” in the same way as Shaw and Gaines [Sha89] (see table in section 1.2)

We can see that conflicts can be solved using simple operations, for example: create a new concept, associate definition to different points of view, “merge” definitions, delete a definition. The choice of an operation must be justified by an argument. In the following tables, we enumerate possible operations to apply in each situation, with examples of typical arguments.

Table 1: D1 is a generalization (specialization) of D2

		Possible operations and typical arguments (arg)
D1 and D2 compatible	1 same concept	<ul style="list-style-type: none"> • keep both D1 and D2 + link them to different points of view argument: the specific characteristics of D2 are useful for a point of view and useless for the other or remove one of them + choose a point of view argument: D1 is incomplete / D2 is hyperspecific (some of its characteristics are redundant) Remark: $union(D1,D2)=D2 / intersection(D1,D2)=D1$ so it is useless to suggest a merge operation
	2 different concepts	<ul style="list-style-type: none"> • create a new concept + create a new term for one of the definition. + create an inheritance link between D1 and D2. argument: the characteristics of D2 that are not in D1 make the specificity of D2.

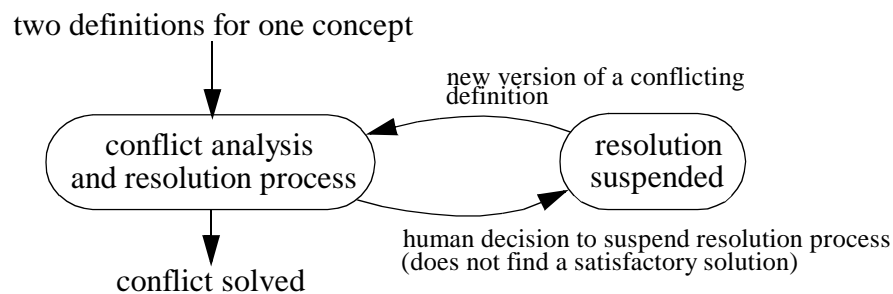
Table 2: D1 is not a generalization (specialization) of D2

		Possible operations and typical arguments
D1 and D2 compatible	3 same concept	<ul style="list-style-type: none"> • keep both D1 and D2 + link them to different points of view or remove one of them argument: D1 and D2 are similar or create a new definition by combining D1 and D2, for example by using $union(D1, D2)$ or $intersection(D1,D2)$ argument: (for union) D1 and D2 are incomplete (not specific enough) or keep both + create a new definition by combining D1 and D2 + link to different points of view argument: D1 and D2 are both interesting, but it would also be useful to have a more “general” definition
	4 different concept	<ul style="list-style-type: none"> • create a new concept + creae of a new term or create a new concept + keep the same term + link to another field argument: D1 and D2 are homonyms

Table 2: D1 is not a generalization (specialization) of D2

D1 and D2 incompatible	5 same concept	same as 3 Remarks: union(D1,D2) is incoherent (empty interpretation) In this case, keeping both D1 and D2 and linking them to different points of view is not a completely satisfying solution, because the knowledge base stays in an incoherent state.
	6 different concept	same as 4

Of course, the proposed resolution process does not automatically lead to an acceptable solution. So, designers may decide to suspend the resolution of a particular conflict and to wait for some new versions of a conflicting definition, as shown on the following diagram.



Keeping track of the decisions: arguments and endorsements

Storing the arguments underlying each operation together with terminological knowledge allows to remember how “final” definitions were chosen, thus avoiding to repeat past reflection. Arguments represent informal knowledge (in the sense of Conklin [Con96]). Arguments are informal knowledge that give a background to operations.

Endorsements act as “checkpoints” in the process. They “mark” situations which are approved by some authority. Even if the knowledge base continues to evolve, they form references. From an end-user point of view, the most interesting definitions are probably not the latest versions but the latest approved versions.

5 Conclusion

In this paper we have presented ConceptIBIS, a formal concept model which is aimed at the collaborative building of terminological knowledge bases. ConceptIBIS provides a multi point of view management support. In addition to its formal part, this model also enables to store informal knowledge that gives information on how the terminological knowledge is built.

Then we described a process to resolve the conflicts that inevitably occur when a group of people is involved in the building of a concept base. This process includes:

- semi-automatic conflict analysis with the help of the concept comparison operation and its resulting differences
- operations to resolve conflicts
- memorization of arguments to justify the choice of an particular resolution operation

- endorsement to express agreement on definitions

We are currently testing the comparison operation, as well as other operations, on a multi-lingual concept base in the field of furniture. The results obtained so far are encouraging.

Meanwhile, we are developing a collaborative system, with a Web interface, for translators and terminologists to easily exchange terminological knowledge. This system currently deals with textual definitions. We are now working on the integration of this ConceptIBIS in this system.

6 References

- [Ber94] Berthet Dominique, Bonjour Michel, De Bessé Bruno, Falquet Gilles, Léonard Michel, Sindayamaze Jeanne, *ConcepTerm "Construction de dictionnaires encyclopédiques multilingues et informatisés"*, CUI technical report, University of Geneva, 1994
- [Bra91] Brachman, R., McGuinness, D., Patel-Schneider, P., Borgida, A. and Resnick, L. *Living with CLASSIC: When and How to Use a KL-ONE-Like Language*, in Principles of Semantic Networks. Morgan Kaufman. Pp. 401-456. May, 1991.
- [Buc97] Buckingham Shum Simon, *Representing Hard-to-Formalise, Contextualised, Multidisciplinary, Organisational Knowledge*, in AIKM'97 Proceedings
- [Buc93] Buchheit Martin, Donini Francesco M., Schaerf Andrea, *Decidable Reasoning in Terminological Knowledge Representation Systems*, Journal of Artificial Intelligence Research, 1993
- [Con89] Conklin Jeffrey, Begeman Michael, *gIBIS: A Tool for All Reasons*, Journal of the American Society for Information Science, 40, pp 200-213, 1989
- [Con96] Conklin Jeffrey, *Designing Organizational Memory: Preserving Intellectual Assets in a Knowledge Economy*, Group Decision Support Systems, 1996
- [Deb89] De Bessé Bruno, Pulitano Donatella, *BD-TERM, un logiciel de gestion terminologique*, ETI, Université de Genève, 1989
- [Die97] Dieng Rose, *Comparison of Conceptual Graphs for Modelling Knowledge of Multiple Experts: Application to Traffic Accident Analysis*, Rapport de Recherche n°3161, INRIA Sophia Antipolis, Projet Acacia, 1997
- [Euz96] Euzenat Jérôme, *Corporate memory through cooperative creation of knowledge bases and hyper-documents*, in Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW '96)
- [Fal91] Falquet Gilles, Sindayamaze Jeanne, Bonjour Michel, Leonard Michel, *F2Concept, un modèle intégrant la description de la compréhension et l'extension des classes d'objets*, CUI, Université de Genève, 1991
- [Gro] Group Decision Support Systems, *The IBIS Manual: A Short Course in IBIS Methodology*, <http://www.gdss.com/IBIS.htm>
- [Kun72] Kuntz Werner, Rittel Horst, *Issues as elements of information systems*, Working Paper No 131, Institute of Urban and Regional Development, University of California at Berkeley, 1972
- [Mar93] Mariño Drew Olga, *Raisonnement classificatoire dans une représentation à objets multi-points de vue*, thèse de doctorat, Université Joseph Fourier, Grenoble, 1993
- [Pul88] Pulitano Donatella, *Création d'une banque de terminologie à l'Ecole de traduction et d'interprétation de l'Université de Genève : BD-TERM*, ETI, Université de Genève, 1988
- [Sha89] Shaw Mildred L. G., Gaines Brian R., *Comparing Conceptual Structures; Consensus, Conflict, Correspondance and Contrast*, Knowledge Science Institute, University of Calgary, 1989
- [Sin95] Sindayamaze Jeanne, *Prise en compte de la compréhension dans les bases de données*, thèse de doctorat, Université de Genève
- [Stu98] Stumpf Simone, *Argumentation-based Design Rationale - The Sharpest Tools in the box*, Working Paper IN/98/01, University College London