# A Reasoning Approach for Conflict Dealing in Collaborative Design

Moisés Dutra[a], Parisa Ghodous[b, 1]

[a]PhD Student, University of Lyon 1.
[b]Head of Collaborative Modeling Team of LIRIS Laboratory, Univ. of Lyon 1.

**Abstract.** In collaborative design process, multidisciplinary virtual teams' integration – involving exchange and sharing of knowledge and expertise – frequently generate a lot of conflicting situations. Different experts' point of views and perspectives, in addition to several ways of communicating and collaborating in knowledge level, make all this process very hard to tackle. Aiming to minimize the appearance of early design conflicts and to solve the rest of them, this paper presents an approach to represent knowledge in design process based on Web Ontology Language (OWL). OWL is structured to be a major formalism for the design and dissemination of ontology information. The use of OWL reasoning is intended to be a consistent way to verify the information given by several experts, trying to avoid redundancies, contradictions and misunderstoods. A prototype, based on the Function-Behavior-Structure design framework, that uses OWL to input data, was built up to validate this approach.

**Keywords.** Collaborative design, ontology integration, reasoning, conflict resolution.

## 1 Introduction

Current trends in collaborative design domain have focused on the increasing needing of teamwork. Complex industrial problems involving different knowledge areas require, even more, heterogeneous virtual teams to collaborate in order to solve them. These teams should exchange their knowledge and expertise, creating a collaborative network.

The enterprises' collaborative design process uses, even more, geographically distributed knowledge, resources and equipment [16]. Virtual team members work in parallel with different tools, languages and time zones. Collaborative engineering proposes to face these problems by reorganizing and integrating the set

[1] Head of Collaborative Modeling Team
LIRIS – Laboratory of Computer Graphics, Images and Information Systems
Bâtiment Nautibus, 43, bd. Du 11 Nov. 1918
69622 Villeurabanne cedex, France
Phone: (33)4 72 44 58 84
Fax: (33)4 72 43 13 12
E-mail : ghodous@bat710.univ-lyon1.fr

of development activities, starting from design early stages [6]. An important part of the collaborative work consists of communication and sharing of data, information and knowledge, among individuals with different points of view and specific objectives concerning their particular domains. Once these points of view and objectives are considered inside the same scope, divergences arisen from this process lead frequently to conflicts.

This paper proposes an approach to attenuate design conflicts. This approach is based on OWL reasoning, where all exchanged data and information will be represented as OWL classes, in order to use an OWL reasoner. To validate this proposal, a little prototype was implemented, taking as scneario a distributed architecure for collaborative design, previously described in [2] and [14].

## 2 Design conflicts and OWL reasoning

### 2.1 Conflicts in collaborative design

A conflict is an incompatibility between two design decisions or between two distinct design objectives [8]. It can be considered that there is a conflict when one or more propositions cannot coexist in the same time at the same design space [4].

Design space: is a multidimensional representation of the design process parameters' set – models, but also design goals, the socio-cultural references that define the designer's work method.

Publication: publishing a proposition means to put together proposed instances of product model and the subpart already instantiated of the design space. This union is only possible if there is no interference between these two sets that means, if all of their elements are coherent.

Attenuation: conflicts can be extremely consumers of resources (development time, budget, materials). That is why it is so important to prevent them, trying to detect them in early design stages. Identify them, categorize them, and notify the different involved parts, in order to put the situation under control as soon as possible. These tree steps are called strategies of conflict attenuation [10][4].

### 2.2 OWL reasoners

The Web Ontology Language (OWL) [12], released as a W3C recommendation in February 2004, is an expressive ontology language that is layered on top of RDF and RDFS. OWL can be used to define classes and properties as in RDFS, but in addition, it provides a rich set of constructs to create new class descriptions as logical combinations (intersections, unions, or complements) of other classes; define value and cardinality restrictions on properties (e.g., a restriction on a class to have only one value for a particular property) and so on.

OWL is unique in that it is the first ontology language whose design is based on the Web architecture, i.e., it is open (non-proprietary); it uses Universal Resource Identifiers (URIs) to unambiguously identify resources on the Web (similar to RDF and RDFS); it supports the linking of terms across ontologies making it possible to

cross-reference and reuse information; and it has an XML syntax (RDF/XML) for easy data exchange.

One of the main benefits of OWL is the support for automated reasoning, and to this effect, it has a formal semantics based on Description Logics (DL). DLs are typically a decidable subset of First Order Logic. They are suitable for representing structured information about concepts, concept hierarchies and relationships between concepts. The decidability of the logic ensures that sound and complete DL reasoners can be built to check the consistency of an OWL ontology, i.e., verify whether there are any logical contradictions in the ontology axioms. Furthermore, reasoners can be used to derive inferences from the asserted information, e.g., infer whether a particular concept in an ontology is a subconcept of another, or whether a particular individual in an ontology belongs to a specific class.

Examples of OWL reasoners [11]:

- F-OWL [5]: an ontology inference engine for OWL based on Flora-2 (an Object-Oriented Knowledge Base Language).
- Euler [3]: an inference engine supporting logic based proofs. It is a backward chaining reasoner and will tell you whether a given set of facts and rules supports a given conclusion.
- Pellet [13] is an OWL DL reasoner based on the tableaux algorithms developed for expressive Description Logics. After parsing OWL documents into a triple stores, the OWL abstract syntax are separated into TBox(axioms about classes), ABox(assertions about individuals) and RBox(axioms about properties), which are passed to the tableaux based reasoner.
- Hoolet [7] is an OWL DL reasoner that uses a First Order Prover to reason about OWL ontologies.
- Cerebra [1] is a product of Network Inference, and its technology provides a commercial grade, robust, scalable implementation of the DL algorithms that use OWL documents in their native form. These algorithms are encapsulated into a run-time engine that is provided as a service to other applications or services and can respond to queries about ontologies from those applications.

The OWL Test Cases document [9] defines an OWL consistency checker as follows: an OWL consistency checker takes a document as input, and returns one word being consistent, inconsistent, or unknown.

But, while consistency checking is an important task, it does not, in itself, allow one to do anything interesting with an ontology. Traditionally, in the ontology and Description Logic community, there is a suite of inference services held to be key to most applications or knowledge engineering efforts. It is imperative that a practical OWL reasoner provide at least the standard set of Description Logic inference services, namely [13]:

- Consistency checking, which ensures that an ontology does not contain any contradictory facts. The OWL Abstract Syntax & Semantics document provides a formal definition of ontology consistency that Pellet uses. In DL

terminology, this is the operation to check the consistency of an ABox with respect to a TBox (this corresponds to being an OWL consistency checker).

- Concept satisfiability, which checks if it is possible for a class to have any instances. If class is unsatisfiable, then defining an instance of the class will cause the whole ontology to be inconsistent.
- Classification, which computes the subclass relations between every named class to create the complete class hierarchy. The class hierarchy can be used to answer queries such as getting all or only the direct subclasses of a class.
- Realization, which finds the most specific classes that an individual belongs to; or in other words, computes the direct types for each of the individuals. Realization can only be performed after classification since direct types are defined with respect to a class hierarchy. Using the classification hierarchy, it is also possible to get all the types for that individual.

## 3 A reasoning approach

The architecture proposed in [2] is based on the Function-Behavior-Structure framework. This framework is built up to take users' requirements, to transform them into formal specifications and to use these specifications to describe product's functions, behaviours and structures (Figure 1). All this process is done by several geographically distributed designer groups, called agencies.

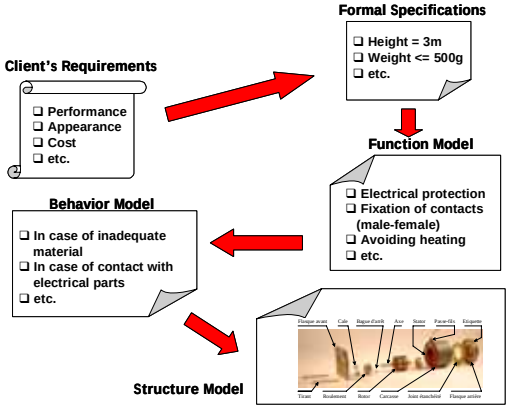


**Figure 1.** Function-Behavior-Structure

Each agency represents a specialty, a knowledge domain. An agency aggregate one or more designers (called agents). Each agent has a private workspace of data and information. In the private workspace, he prepares his propositions before submit them to other agents. The interaction with other agents is provided by an

agency shared workspace (ASW). The ASW has the same structures present in agent private workspace, but in a higher level, because all data and information contained in there were previously agreed among all concerning agents. In the same way, communication inter-agency is provided by a project shared workspace [15], a global workspace to store the final design models (Figure 2).

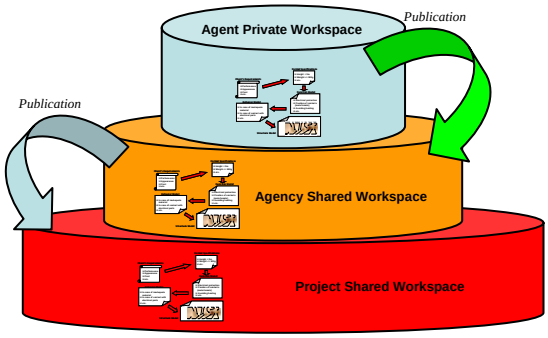


**Figure 2.** Shared Workspaces

Our proposal consists of creating an intermediate step when publishing to higher levels. The approach is to execute an OWL consistency checking in every proposition to be published. This intends to guarantee the coherence of the information being passed through. The goal is to take advantage of the OWL reasoning to prevent early conflicts. For example, it can be defined an OWL superclass to represent a metal alloy, named ClassA. ClassA represents structures constructed using aluminium-cooper alloy. ClassA may determine that its weight property cannot have value higher than 100g. So, for all subclasses representing structures of aluminium-cooper alloy, this stands as a rule to be followed from now on. If a ClassB intends to extend ClassA, its weight property must necessarily be set to less or equal 100g, otherwise, it will not be published.

## 4 Prototype

A prototype has been built up in order to validate the proposed approach. Providing a computational environment, it intends to be an infrastructure tool through which several experts can collaborate in order to achieve a product design. Collaboration in this context comprises interaction among different knowledge areas. In an effective and collaborative environment, experts came from different areas can integrate their knowledge in a simultaneous way, looking for an expressive gain of productivity.

Experts have their own way to see and to understand the subject, according to their duties. Therefore, providing mechanisms of data representation to create a “common language” understood by the experts was the way chosen to detect specification conflicts.

An electrical connector design, as depicted in [6], was chosen as scenario to this prototype. The work required to deploy this connector is decomposed in

several subsystems, corresponding to different disciplines (mechanical, electrical, thermal, etc.). The goal is to simulate the design process of this trying to identify potential conflicts, by using OWL reasoning. Figure 3 shows the connector's overall structure and his components.

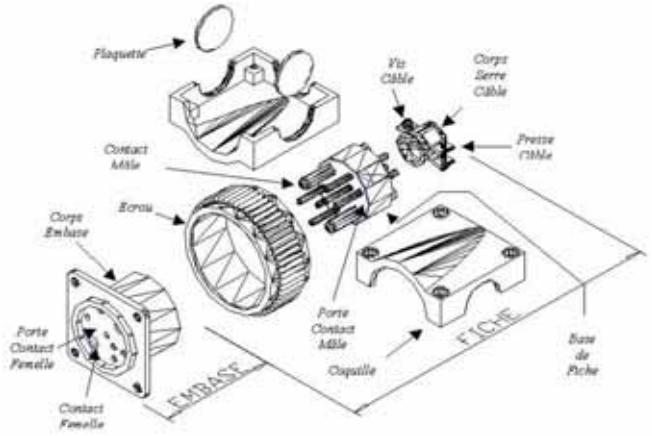


**Figure 3.** Electrical Connector Design

The prototype is based on John Gero's Function-Behavior-Structure design framework [14]. According to this framework, a product structure is obtained starting from function and behavior models related to the product. Every structure model is derived from one or more behavior models. Behaviors are derived from function models and every function will be defined from one or more formal product specifications. Hence, as these models are interrelated, the main goal here is to formalize product requirements, turning them into formal specifications (OWL classes).

The prototype's internal structure comprises three roles: system administrator; specialists, called agents; and specialist groups, called agencies. The system administrator is the project manager. He is in charge of project creation, definition of objectives, date delimitation, attribution and management of agents and agencies. Attributions given to agents and agencies include right access and system using access. Actors are agents (experts) and agencies (expert groups) which are involved in the product design.

Figure 4 shows a potential conflict situation. Inside the Design Agency can be seen two experts, a mechanical engineer and an electrical engineer. Both of them disagree about a specific piece height, what generates a conflict.

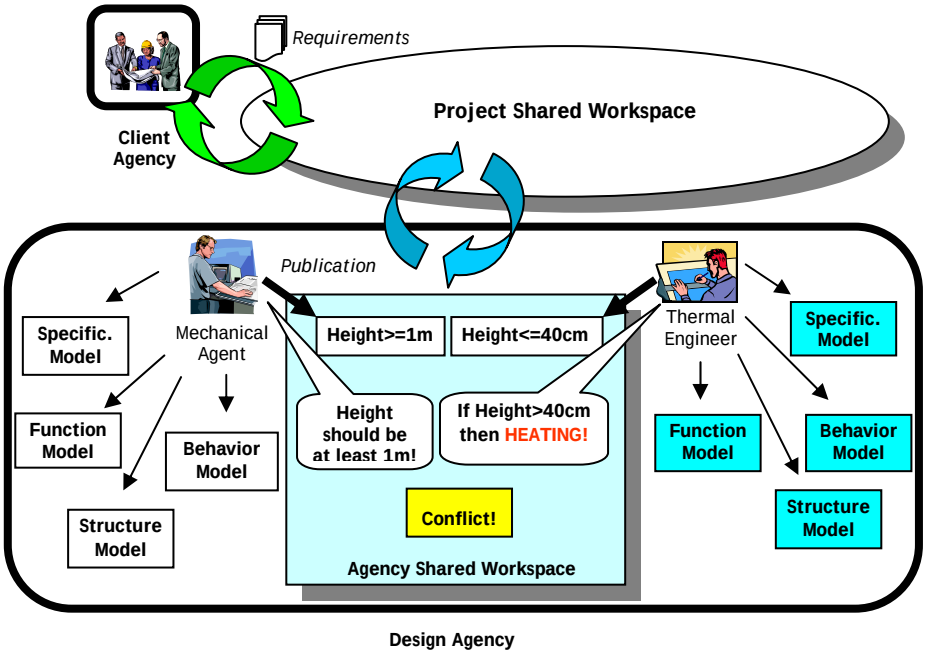


**Figure 4.** Potential Conflict Situation

In this case, the conflict was avoided because before the second agent publish his proposition, he was informed by the system that the content to be published was not consistent neither coherent. So, in this case he must necessarily to redo his work or try to negotiate with the other involved agent.

## 5 Conclusions and perspectives

Representing knowledge in OWL offers a reasonable trade-off between expressibility and decidability, witch when used to verify product specifications in collaborative design may fit as an efficient conflict attenuator. Nevertheless, two important limitations remain: efficient reasoning on real-world ontologies containing a large set of individuals is still a challenging task and OWL cannot be used efficiently to model certain application domains.

To attenuate these problems, we intend to model applications by using EXPRESS language (ISO 10303-11), an industry standard conceived to provide information and data exchange and system interoperability. This step intends to give robustness to the system. For such, to achieve such a scenario, a conversor “EXPRESS to OWL” is being built up, to keep the advantages of the OWL reasoning.

Moreover, next steps comprise implementing the agents’ negotiation process and also to improve the OWL reasoning to work with the case-based approach, profiting from the entries previously put into the knowledge base.

## 6 References


[1] Cerebra. Available at: http://www.semtalk.com/cerebra_construct.htm. Accessed on: March, 2007.

[2] Dutra M, Slimani K, Ghodous P (2007) *A Distributed and Synchronous Environment for Collaborative Work*, submitted to the Integrated Computer-Aided Engineering journal (ICAE), ISSN 1069-2509, vol. 14, IOS Press, Amsterdam, The Netherlands, 2007.

[3] Euler proof mechanism. Available at: http://www.agfa.com/w3c/euler/. Accessed on: March 2007.

[4] Ferreira da Silva C, Médini L, Ghodous L (2004) *Atténuation de conflits en conception coopérative* [in french]. In *15èmes journées francophones d'Ingénierie des Connaissances* (IC'2004), Nada Matta ed. Lyon. pp. 127-138. University Press of Grenoble, Grenoble, France, ISBN 2 7061 1221 2, 2004.

[5] F-OWL. Available at: http://fowl.sourceforge.net. Accessed on: March 2007.

[6] Ghodous P (2002) *Modèles et Architectures pour l'Ingénierie Coopérative* [in french]. Habilitation Thesis,University of Lyon 1, Lyon, France, 2002.

[7] Hoolet. Available at: http://owl.man.ac.uk/hoolet/. Accessed on: March 2007.

[8] Klein M (2000) *Towards a systematic repository of knowledge about managing collaborative design conflicts*. In Gero J (ed.), Artificial Intelligence in Design '00, Boston. Dordrecht: Kluwer Academic Publishers, pp. 129-146, 2000.

[9] Kalyanpur A (2006) *Debugging and Repair of OWL Ontologies*. PhD Thesis, Faculty of the Graduate School of the University of Maryland, USA, 2006.

[10] Matta N, Corby O (1996) *Conflict Management in Concurrent Engineering: Modelling Guides*. Proceedings of the European Conference in Artificial Intelligence, Workshop on Conflict Management, Budapest, Hungary, 1996.

[11] Mei J, Bontas EP (2004) *Reasoning Paradigms for OWL Ontologies*. Technical Report B-04-12, Institut für Informatik, Freue Universität Berlin, Germany, November 2004.

[12] OWL Web Ontology Language Overview. Available at: http://www.w3.org/TR/owl-features/. Accessed on: March 2007.

[13] Sirin E, Parsia B, Grau BC (2004) *Pellet: A Practical OWL-DL Reasoner*. Proceedings of 3rd International Semantic Web Conference (ISWC2004), Hiroshima, Japan, November 2004.

[14] Slimani K (2006) *Système d'échange et de partage de connaissances pour l'aide à la Conception Collaborative* [in french]. PhD Thesis, University of Lyon 1, Lyon, France, September 2006.

[15] Sriram RD (2002) *Distributed and Integrated Collaborative Engineering Design*. Savren, 2002. ISBN 0-9725064-0-3.

[16] Xie H, Neelamkavil J, Wang L, Shen W and Pardasani A (2002) *Collaborative conceptual design - state of the art and future trends.* Proceedings of Computer-Aided Design, vol. 34, pp. 981-996, 2002.