Theme:

Title:

Regulation checking in a Virtual Building

Author(s): S. Maïssa[1,2], JP.Frachet[2], JC. Lombardo[1], M. Bourdeau[1], S. Soubra[1]

Institution(s): (1) Centre Scientifique et Technique du Bâtiment (CSTB)

(2) Ecole Nationale Supérieure des Arts et Métiers(ENSAM), Institut Image

E-mail(s): Sandrine.maissa@cstb.fr

Abstract: *Use of virtual buildings is a good way to get a concrete idea about future projects. It may allow avoiding errors discovered after the construction start and thus expensive to correct. In the near future, virtual environments should allow taking into account the whole constraints linked to the construction project (e.g. architectural, technical, regulations, financial, etc.). This is the purpose of CSTB's Enriched Virtual Environments project (EVE), which aims to offer users the possibility to couple 3D representation of the built environment with physical simulations (thermal, acoustic, etc.) within a Virtual Building. Moreover, EVE integrates a code compliance checking module, which provides users an easy and quick code compliance access. This access to regulation documents enables the user to query in an intuitive way the regulation database that applies to his project, and more particularly according to his interest (fire security, accessibility, etc.). This paper focuses on possible solutions to manage this last issue, which rely on the building data structure and the database containing French building code compliance and documents. The main objective for this work is to conceive a matching to bind these two concepts. Data structure is based on the Industry Foundation Classes (IFC), and some tools have been implemented upon these classes, to allow code compliance access. The French regulation database is the CD-REEF one.*

Keywords: *Code compliance accessing, IFC, Virtual building, regulation database.*

Introduction

For some years now, Virtual Reality (VR) technologies have proved their efficiency for virtual prototyping in the building design phase (1)(2)(3). Virtual prototyping offers to project stakeholders the possibility to analyse their building with respect to various criteria (thermal, acoustic…) during the early design phase thanks to numerical simulations (4). This allows to save time on design schedules, to avoid late discovery of inconsistencies in the building design, and thus to improve the whole quality of the finished building and decrease its cost. Picking up where Computer-Aided Design (CAD) software leaves off, virtual prototyping software simulates not just the way things look but also the way things work… Existing virtual prototyping software mainly deal with physical information issued from numeric simulations. Nevertheless, code compliance checking has an important place during the design phase and is being studied by some research projects. For instance, the EEC ISTFORCE[1] project (5) has a Code Compliance Checking of Architectural Design Solution integrated in a collaborative environment. This module is based on a knowledge-based code checking service. It helps to check if a project complies with a regulation by automating a part of the needed verifications. Another similar project is The Construction and Real Estate Network project (CORENET[2]), a major Information Technology (IT) initiative undertaken by Singapore Ministry Development to re-engineer the business processes of the Construction Industry. It provides an automated code checking based on intelligent IFC2x objects, which allows automated approval of building plans over the Internet. In the United States, the Department of Civil and Environmental Engineering of Stanford University (6) has launched a research project to develop a formal information infrastructure. That will enhance the access and retrieval of government regulations. In the information service framework, government regulations will be made available on-line, and tools

[1] ISTFORCE see: http://www.istforce.com
[2] CORENET see: http://www.corenet.gov.sg

will be provided to locate, merge, compare, and analyse the information. Most of these research are based on automated code compliant checking. Currently, even if this kind of checking is convenient, results are rarely exhaustive, and thus, that can imply oversights. This paper introduces a complementary way to check a project against code compliance. It offers an access to a complete code compliance database within a virtual building walkthrough. It is the user's responsibility to check whether his project complies with the regulation or not.

This paper describes the involved technologies to select and retrieve from regulation database information the user expects. The firsts sections describe the French regulation database (Reef) and the data structure used to represent the built environment (IFC). CSTB's virtual prototyping platform EVE, in which our work is integrated, is then introduced. Details on the implementation are given in section 4. Results are finally discussed.

French regulation database for Building and Construction (B&C).

Regulation tools

The REEF is the compendium of laws, standards and technical rules that apply in the B&C sector in France. REEF is edited and commercialised by CSTB. This collection is made of twenty-five books. These volumes provide the Building professionals all the necessary information for design, execution, control and maintenance of buildings. In the 90's, CSTB have conceived and edited an electronic version of this product to facilitate regulation accesses. The CD-REEF[3] centralizes the whole building code compliance within a documentary database. Document types are: code compliance, schemas, tables or formulas. CD-REEF software is structured in two databases. The first one contains referenced documents, and the second one is a bibliographic database used to reference, catalogue and to index in a manual way documents. Two kinds of query are available: access to a document or to a specific part of a document, according to needs.

Document access

To provide an efficient and easy way to access documents; they have been hierarchically sorted according to several contexts. Theses contexts are: building components (door, stair, beam, etc.), demands (accessibility, fire security, hygiene, etc.), administrative constraints (private transactions, building certificates, insurance, etc.) and functions (one-family dwelling, multi-storey dwelling, industrial facilities, etc.). Other ways to access documents exist, but they are not relevant to this work. In fact, the most relevant way of accessing document in accordance with the work described in this paper is the access through building products. It can be compared with the IFC data structure more accurately than the others.

Document part access

An indexation allows the user to get only the part of document that fits the query (i.e. the smallest referenced document part, that is composed of few lines). This indexation is generated thanks to a linguistic application software called SyLex (7). This tool scans the whole code compliance and indexes each significant word. A human validation is required to add new terms in the index after having verified their relevancy. Results are stored in a table that links the key word with the piece of code text. This table is huge, and computation time to find the pertinent text parts is long. In consequence, current CD-REEF® version only allows the user to use this form of query after having pre-select documents thanks to query of the previous types (i.e. contexts).

Industry Foundation Classes, a data structure to represent buildings

IFC overview

To enhance the possibility to store and exchange architectural projects, most of the architectural CAD tools' manufacturers have grouped themselves in an International Alliance for Interoperability (IAI[4]). With other partners, they designed an object-oriented data model to represent these projects: the Industry Foundation Classes (IFC). Modelled using the STEP standard, IFC entities allow to represent the semantic of building elements with their relations and properties through inheritance. Because CAD

[3] CD-REEF: see http://www.cstb.fr/app/reef/a_propos.htm
[4] IAI: see http://www.iai.org.uk

software editors (8) are involved in the definition of this standard, IFC are supported in most of CAD tools. The IAI released in October 2000 the 2.x version of IFC, but its implementation is not yet commercially available. Thus, this work is based on IFC 2.0.

According to the IFC 2.0 formalism (9)(10), an architectural project is described by an *IfcProject*. It handles one or more *IfcSite*. Each site defines a location (and possibly a geometrical representation of the ground) where one or more buildings (*IfcBuilding*) are to be built on. Each building is made of several storeys (*IfcStorey*). The *IfcStorey* keeps a link on the building elements (*IfcBuildingElement*) that compose it. The *IfcBuildingElement* is an abstraction used to describe all the building components like walls (*IfcWall*), doors (*IfcDoor*), windows (*IfcWindow*), stairs (*IfcStair*), beams (*IfcBeam*), columns (*IfcColumn*), etc. *IfcBuildingElement* have common attributes as, for instance, a geometry (*IfcGeometry*), a location relative to the *IfcBuildingStorey*, a name, a unique ID, etc. and some specific attributes. The entities defined in the IFC model handle properties in the same way and provides the user the ability to create and manage its own property set. As an example, an *IfcDoor* has a link on a *Pset_DoorCommon* property set which defines the door features (i.e. the boolean *IsExterior* specifies whether the door opens on the exterior or not, the length *NominalHeight* gives the nominal door height, etc.).

Using IFC

Potentially, the IFC description of a project should give a lot of information that can be used to drive physical simulations or code compliance checking. However, as many of the IFC entities or properties are optional, it is extremely rare to have a complete model. Moreover, some information are missing in the data model itself. For instance, the building function is very important for code compliance checking, as one-family dwelling, industrial facilities or public building do not have the same constraints regarding to the French laws. On the other hand, even if the needed information is present in the model, due to the IFC model complexity, it may take rather a long computation time to retrieve it. For these reasons, it is necessary to have a way to store user data on the model itself.

Adding data in IFC data structure

To make IFC a usable data model able to support numerical simulations and to perform code compliance checking, we have to enrich the model by adding specific technical data. Some research deal with the use of IFC in simulation tools and potential extensions of IFC. Weise & al. suggest enhancing the IFC model by adding an extension for structural engineering domain (11). This was reached by adding a minimal set of new classes and using property set objects. Bazjanac also uses IFC in simulation tools (12). He describes the process of definition and addition of new classes to IFC. From our point of view, adding new classes to the IFC 2.0 data model is not the good way of adding information to the model unless it is done by the IAI (which leads to another version of the IFC). Modifying the IFC classes should lead to a new data structure no more IFC compliant and thus not supported by standard CAD tools. The storage of new information is allowed in the IFC 2.0 specifications by the addition of new property set on the entities. Thus, we choose to store our needed information as new properties (*IfcSimpleProperty*) and properties set (*IfcPropertySet*).

EVE, a virtual prototyping framework

CSTB’s Enriched Virtual Environments project (EVE) aims to design and develop a virtual prototyping tool that gathers both physical and regulation information. EVE gives an access to this information through a 3D representation of the building project. Code compliance access is offered within a virtual walkthrough where the user selects the building elements he is interested in.

EVE is a software platform dedicated to the construction sector (including buildings and urban scences). It is issued from a CSTB strategic project that begun in 2000 for an initial length of 5 years.

EVE’s aims

EVE is designed to provide users with a unified and coherent environment to evaluate a project during its design phase. To achieve this goal, EVE is based on a central data model used to run several physical simulations and to access to code compliance. EVE offers a VR front-end to visualize both the architectural project and the simulation results. Such a VR interface eases the communication between different actors (designer, client, local authorities, residents, etc.) by giving them a common and intuitive 3D representation of their project.

Four types of information are provided by EVE:

- Geometry and semantic information on the project.
- Physical information about the building interior thanks to several numerical simulations (acoustic, thermal, lighting, fire security, etc.) to validate the design technical options;
- Physical information about the building environment (air quality, outside acoustics, etc.) to test the project integration within its neighbourhood;
- Regulation issues through an access to a code compliance database to allow a quick and easy regulation checking.

EVE is extensible through a plug-in mechanism and runs on different hardware platform.

Software Architecture overview

EVE is made of a main application and several modules. Modules (or plug-ins) can be loaded and unloaded at runtime. EVE's heart ensures communication between the plug-in.

EVE's main features are the following:

- **Extensible software architecture.** Through its plug-in mechanism, it is straightforward to add new functionalities to EVE. Communication between modules is performed thanks to signal broadcasts. Events (e.g. a new model has been loaded, a model has been modified, an object has been selected…) are sent to registered modules. Each module can react to these events. Several modules exist in EVE, some modules help to manage the data while others deal with simulations. The IFC module is an example of a data handler module. It supports the IFC model and provides some browsing and multi-selection capabilities. Acoustic and thermal simulations are integrated in EVE as simulation modules. Virtual acoustic plug-in provides both acoustic simulation and perceptive rendering of simulated results. It allows a user to evaluate levels of isolation by listening to familiar sounds in the future building according to construction materials and room configurations he has chosen. The simulation code comes from a tool named ACOUBAT-Son that has been developed in CSTB (13). Thermal simulation module offers to assess both energy consumption of a building and comfort conditions in each room. It is based on a well-known package called TRNSYS (14).
- **Central data model.** As far as building projects are concerned, EVE is based on an IFC representation of the building. EVE uses CSTB's C++ library to handle the IFC data model. This Open Source library[5] provides an early binding of the STEP schema that describes the IFC and has been automatically generated from EXPRESS. Each IFC object is mapped to a C++ class, including the access to the object's attributes. The IFC library is also able to read and write STEP Neutral Files (SPF). Architectural project modelled with CAD tools can thus be loaded in EVE, modified by EVE (i.e. by adding needed properties) and then re-exported to CAD tools.
- **Interactive virtual representation of the built environment.** The geometric representation contained in the IFC model is processed to obtain a polygon representation suitable for real-time rendering. A link between this representation and the original IFC entities is kept, so the semantics are not lost. When the user selects an object in the 3D rendering window, its features can be shown. Simulations and code compliance checking are driven from this 3D interface. The results are also displayed there allowing the user to understand the link between the model and the results simulation produces (15).
- **Scalability and portability.** EVE is intended to run on on various hardware platforms, from the laptop to the Reality Centre. EVE is written in C++. It uses Qt 3.0[6] for the Graphical User Interface (GUI) and OpenGL Optimizer[7] for the 3D rendering. The extraction of the geometry from the IFC is performed thanks to OpenCascade's[8] STEP Part 42 binding. EVE currently runs on **IRIX**® and **Windows**® platforms.

[5] This library is available for download on http://cic.cstb.fr/ilc/CSC/OpenSource/IFC20.htm
[6] QT (Trolltech), see http://www.trolltech.com
[7] OpenGL Optimizer (SGI), see http://www.sgi.com/software/optimizer/
[8] Open CASCADE, see http://www.opencascade.com

Code compliance checking implementation

EVE's code compliance checking module uses semantic information from an IFC representation of an architectural project to retrieve information from the CD-REEF. Our aim is double:

- Give to the user an access to the whole compendium of the regulation code through an intuitive interface;
- Give an easy access to the minimal but exhaustive set of information that answers the query, taking into account the specificities of the studied architectural project.

To achieve this last goal, the building model has to be as complete as possible. It means that the user is allowed to add some specific information on building elements. This is done thanks to the IfcPropertyEditor module.

IfcPropertyEditor

This module manages properties (*IfcProperty* and *IfcPropertySet*) on IFC objects. It allows:

- To fill in or edit the existing IFC properties. Architects that design the original project seldom fill in optional properties, thus the user have to do it.
- To create, edit and delete new properties. These new properties are useful to bypass the IFC data model lacks or to store modules specific information.

For instance, the building destination (one-family dwelling, school, industrial facilities, etc.) impacts all the code compliance checking. The regulation applying to **public buildings** is far more complete than the one applying to the **one-family dwelling**. The element purpose has to be taken into account as well. Query results are different if a door is a **cave**, or an **entrance** one.

IfcPropertyEditor is used by the compliance checking to store needed information on the building elements.

General processing of a code compliance request

While the module is active, the selection of an object starts the code compliance checking. A first set of documents is returned according to the selected element type thanks to the static matching. The user can specify his interest (fire security, safety of use, accessibility, protection against noise, etc.), which is taken into account to retrieve specific parts of documents thanks to the dynamic matching. Returned code compliance documents are sorted according to seven document types, (i.e. *Documents Techniques Unifiés (DTU*[9]*), Textes législatifs et règlementaires*[10]*, Solutions techniques*[11], etc.). Each document is split up in many parts that are composed of few lines; these parts, named Information Units (IU), are emphasized according to their relevancy. A GUI offers the possibility to browse only these IU. Finally, chosen documents are displayed thanks to an HTML browser.

Matching Methodology

The aim is to establish a link between IFC objects and those handled by the REEF regulation database. For instance, a door is described by its material, its size, its type, the two rooms it connects, etc. These attributes have to be transcribed without losses from the building model into a query to the database. Two complementary steps are needed to achieve this: the static and the dynamic matching.

Static matching

The **"static matching"** step relies on a translation of the IFC hierarchical data structure into the REEF context hierarchy of building products. It is thus independent of the studied project; it is the reason why it is called static.
Static matching is done thanks to a database made of three tables (see Fig. 1.).
The first table (Ifc-elt) contains information about IFC objects and their hierarchy:

- Identifier;
- Object label;
- Depth in hierarchy;

[9] *Documents Techniques Unifiés (DTU)*: Unified Technical Documents
[10] *Textes législatifs et règlementaires*: Code compliance
[11] *Solutions techniques*: Technical solutions

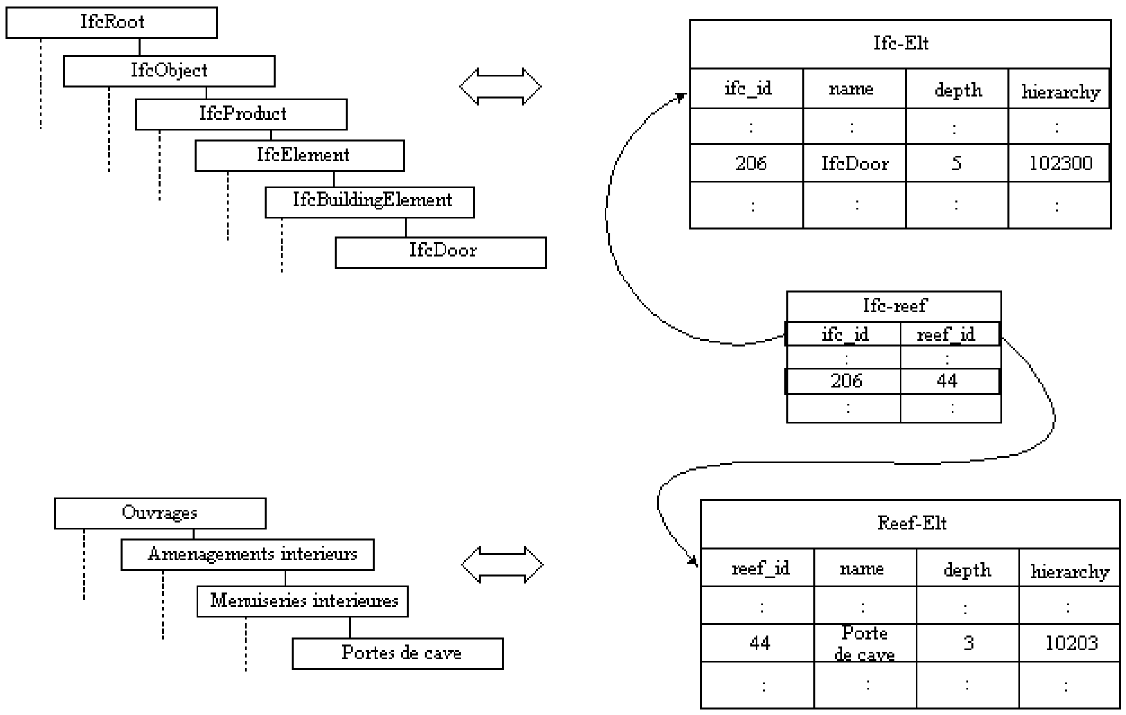

Fig. 1: Static Matching overview

- A number to allow deducing its place in the hierarchy (i.e. to know their "fathers").

This table is automatically built thanks to a text file that represents the IFC hierarchy. It depends on the version of the IFC.
The second table (Reef-elt) describes the REEF context of building products. It contains the same fields that the previous one. This table belongs to the regulation database table set.
Finally, the third table (Ifc-Reef) establishes the link between the two previous ones. It contains:

- IFC identifier
- REEF identifier.

This table is manually made; it depends on releases of IFC and CD-REEF database.
Code is implemented to ease the migration of the matching database by writing a report of what has been changed from a release to another. Unfortunately, the migration cannot be achieved without a user intervention. The building of these tables is made once for given releases of the IFC and REEF. It is a pre-process.
The static matching takes as an input the type of an IFC object and returns a set of related documents. This set can be huge (i.e. a query on *IfcDoor* gives 70 regulation documents).

Dynamic Matching
The **"dynamic matching"** step relies on the IFC model richness, and more particularly on information contained (or added) by IFC object attributes and properties. It uses all the available information on the studied project to restrict the set of previously returned documents, and thus it is computed at run time.
Once the selected IFC object is passed to the dynamic matching, its properties attributes and relations are parsed to extract the needed information. The found information is stored in specific *IfcPropertySet* corresponding to each element types. Information that can describe a

Property Names	**Value examples**	**Manual (M)/ Extracted (E)**
Room1 Function	Cave	M or E_S
Room2 Function	Corridor	M or E_S
Opening type	Sliding	E_D
Material	Wood	M or E_D

Tab. 1: Properties required for a door

door from a regulation point of view is shown in Tab.1. The third column expresses the way data are got. This depends on the completeness of the model. If the model has been completely fulfilled, then most of the attributes can be extracted (E in table), else they can be manually filled (M). Two ways of extracting information are possible. The simplest is when the required information is directly stored on the object as an attribute or as a property (E_D). However, the IFC model allows getting more information by gathering information that is dispatched on related objects (E_S). For instance, an IFC door keeps a link on the wall that contains it. The *IfcWall* keeps a link on all the spaces it delimits. The *IfcSpace* can have an attribute that describes its usage. Following all these links leads to know what is the function of the door (i.e. cave door, entrance door, etc.).

All the found properties are then shown to the user in a GUI. The user selects which ones are relevant to his query and the database is consulted.

The dynamic matching takes as an input the selected object, the set of documents issued from the static matching and the whole IFC model. After a user validation, it returns the set of parts of documents where he will found the answer to his query.

Implementation tips

The CD-REEF is implemented thanks to Microsoft Access. As EVE is a multi-platform software, we had to translate these databases in a more portable format. We chose Mysql[12], which is available on the all targeted platforms. The code compliance documents have been translated from RTF to HTML and are accessed by EVE through the HTTP.

Gathered information (either in a manual way, thanks to a GUI, and/or to previously described software routines) is stored as new properties thanks to the *IfcPropertyEditor*. This information is grouped in a set with a recognizable name. The Code Compliance Checking module creates and fills *IfcPropertySet* named *Pset_**elementType**REEF*, (i.e. *Pset_**Door**Reef*, *Pset_**Window**Reef*, *Pset_**Beam**REEF*, etc.). These properties are IFC compliant and thus, they are kept even if the IFC model is modified with a standard CAD tool. It means that they will be available for next sessions of code compliance checking.

Discussion

Main architectural CAD tools offer IFC 2.0 import/export facilities. We are using. ArchiCAD[13] 6.5 (from Graphisoft). It provides functionalities to fill in some properties of building elements according to the IFC specifications. Unfortunately, only a subset of the building element's properties is editable. Thus, even if the architect whish to create a complete model, he cannot fulfil all the needed properties. The user need to manually gives missed information.

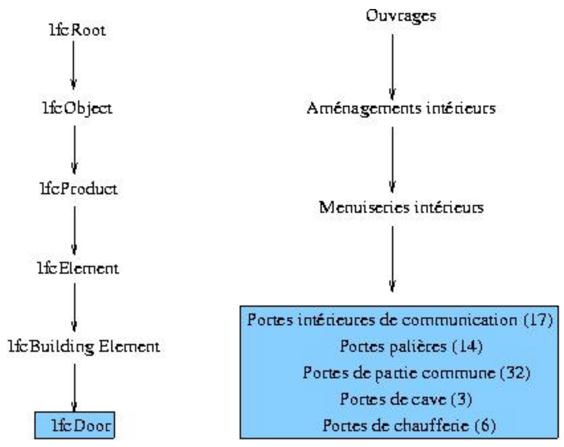

Fig. 2: comparison of the two hierarchical structures

Static matching Limitations

The two structures we need to link are very different. Fig.2. shows a matching example for the door element. The main problem is that there is no one to one relation between the IFC elements and the building products described in the CD-REEF:

- Not every IFC element has got an image in the building products;
- Reciprocally some of the building products cannot be linked to an IFC element;
- An IFC element can match several building products;
- A building product can be linked to more than one IFC element.

The worst case is when the IFC element has no corresponding building product. In this case, the static matching does not return any document, and the dynamic matching has to parse the whole database.

[12] MySQL see: http://www.mysql.com/
[13] ArchiCAD : see http://www.graphisoft.com/products/architecture_and_design/graphisoft_archicad/

Dynamic matching Limitations

The dynamic matching is a very powerful tool. It allows retrieving the existing information dispatched in the model. Its main limitation is that this information is seldom present in the original model and thus need to be manually filled in. As the information is stored in an IFC 2.0 compliant way, this work has only to be done once. If this work has not been done, the code compliance module is not able to select the minimal set of regulation texts that fit the query. It returns a too large number of documents. The user has to find himself the requested information in the given set.

Conclusion

The work presented in this paper aims to enhance a virtual prototype of an architectural project by providing an access to the French code compliance database CD-REEF. The user can query the database by selecting building components during a walkthrough in a virtual representation of the project. A methodology to restrict the returned set of documents has been presented. It is based on the use of the semantic information and properties available from the IFC 2.0 model of the building. Once the properties are filled in, they are stored in the IFC project in a compliant way and are thus reusable.

Ongoing works aim to highlight the building element according to the user interest. For instance, if the user is concerned with disabled people accessibility, involved building elements (i.e. doors, stairs…) have to be highlighted to guide the user.

Literature

1. S. Soubra, F. Coudret. *Virtual Reality in the Construction Industry – Where does it fit in?* CIB, Technology innovation and integration in Construction - University of Florida. February 1998.
2. S. Soubra, F. Coudret, J. Duchon, L. Da Dalto. *Virtual Design and Construction.* Proceedings VRIC, Laval Virtual 2001.
3. W. Thabet. *Design/Construction Integration thru Virtual Construction for improved Constructability*. Berkeley-Stanford CE&M Workshop. 2000. http://www.ce.berkeley.edu/~tommelein/CEMworkshop.htm
4. T. Hodgson, *Virtual Prototyping*. Comptek Federal Systems, Inc., Dahlgren, VA, January1998. http://www.nswc.navy.mil/cosip/feb98/vi0298-3.shtml
5. P. Katranuschkov, R.J. Scherer and Z. Turk. *Intelligent services and tools for concurrent engineering – An approach towards the next generation of collaboration platforms.* Itcon (Electronic Journal of Information Technology in Construction). http://www.itcon.org, 2001.
6. S. Kerrigan, G. Lau, L. Zhou, G. Wiederhold, K.H. Law. *Information Infrastructure for regulation Management and Compliance Checking*. National Conferences on Digital Government Research. pp. 167-170. Los Angeles, May 21-23, 2001.
7. P. Constant. *L'analyseur linguistique sylex*. In 5ème école d'été du CNET. 1995.
8. Graphisoft, IFC Reference Guide. http://www.graphisoft.com/products/ifc_support/
9. P. Poyet and J.L. Monceyron. *Les classes d'objets IFCs Finalités et mode d'emploi*. Technical Report 2986, Centre Scientifique et Technique du Bâtiment, October 1997.
10. R. Billon. Comprendre les concepts des IFC, Décrire son projet en vue des échanges. Plan Urbanisme Construction et Architecte. September 1999.
11. M. Weise, P. Katranuschkov and R.J.Scherer. *A Proposed Extension of the Project Model for Structural Systems*. "Product and Process Modelling in Building and Construction", Proc. ECPPM 2000, pp. 229-238. Lisbon, Portugal, 25-27 September 2000.
12. V. Bazjanac. *The implementation of Industry Foundation Classes in Simulations Tools for the Building Industry*. "Building's 97 Simulation Conference", Prague, Czech Republic, September1997.
13. J. Maillard, J. Martin. *Acoubat-Son, Notice technique*. CSTB. Juillet 2000.
14. TRNSYS. *A transient Simulation Program*. Solar Energy laboratory. University of Madison-Wisconsin.
15. Maïssa, S. and Lombardo, J.C. *Virtual Reality for Scalar Field Visualisation in An Architectural Environment: Using an Avatar.* In Proceedings of GTRV. Conf. First French-British International Workshop on Virtual Reality: Virtual Environment. Brest 2000.