ELSEVIER

Automation in Construction 10 (2000) 79–99

AUTOMATION IN CONSTRUCTION

www.elsevier.com/locate/autcon

An industry foundation classes Web-based collaborative construction computer environment: WISPER

I. Faraj [a,*], M. Alshawi [b,1], G. Aouad [b,2], T. Child [c,3], J. Underwood [d,4]

[a] *Centre for Construction IT, Building Research Establishment, Bucknalls Lane, Garston, Watford WD2 7JR, UK*
[b] *Department of Surveying, University of Salford, Salford M5 4WT, UK*
[c] *Engineering Technology, Technology House, Salford, UK*
[d] *CAE Research Group, School of Civil Engineering, Leeds University, Leeds, UK*

Accepted 20 August 1999

Abstract

Collaborative working in construction is becoming a reality as many activities are performed globally with actors based in various geographical locations. This paper discusses the development and implementation of a collaborative working environment for construction at the University of Salford which is known as Web-based IFC Shared Project EnviRonment (WISPER). The environment is based on a three tier architecture, where user interfaces, business logic and database are kept separate. A Web and Industry Foundation Classes-based (IFC-based) distributed computer integrated environment has been developed. This environment supports design (CAD), visualisation (VR and Drawing Web Format — DWF), estimating, planning, specifications and supplier information. WISPER enables project information to be exchanged through a STEP Part 21 file and shared through the IFC database. Meanwhile, a set of Web pages allows for remote interaction, as well as access to and the distribution of applications. This provides great flexibility and portability, thereby enabling construction professionals to contribute as well as to perform and manage their own activities.

Keywords: IFC; Collaborative engineering environment; Distributed integrated environment

1. Introduction

The current situation within the construction industry is that many practitioners including designers, engineers and suppliers are involved in a one-of-a-kind project which requires coordination. A typical construction project consists of a number of organisations and teams which are brought together for the duration of the project to form a so-called ''virtual enterprise''. This ''virtual enterprise'' often contains units that vary in terms of physical location and computer platforms but which need to work collaboratively and share the same project data. Such issues need to be considered carefully when computer environments are developed.

The open standards and wide accessibility of the Internet mean that it is fast evolving into a powerful environment for supporting distributed and collabo-

* Corresponding author. E-mail address: faraji@bre.co.uk (I. Faraj).
[1] *E-mail address*: m.alshawi@surveying.salford.ac.uk (M. Alshawi).
[2] *E-mail address*: g.auoad@surveying.salford.ac.uk (G. Auoad).
[3] *E-mail address*: t.m.child@surveying.salford.ac.uk (T. Child).
[4] *E-mail address*: cenjun@civil.leeds.ac.uk (J. Underwood).

0926-5805/00/$ - see front matter
PII: S0926-5805(99)00038-2

rative group work. Originally, the Web started as a static, two-tier, client–server, unidirectional environment for the publishing and broadcasting of electronic documents. However, demands for dynamic content has begun to change that. The Web is being transformed from a form of publishing technology into a full-blown client–server medium with the potential to run line-of-business applications and to deal with the complex requirements of multistep business-to-business and consumer-to-business transactions [22,23].

The technological requirements for the development of a distributed computer integrated environment based on three tier architecture and Industry Foundation Classes (IFC) product model was described in [14]. This paper reports on the implementation of this type of distributed computer integrated environment, namely Web-based IFC Shared Project EnviRonment (WISPER), within the construction domain. Many emerging technologies and standards are brought together in this project. These include the IFC, Web technology, Common Object Request Broker Architecture (CORBA) and Virtual Reality Modeling Language (VRML). Standard off-the-shelf software is also used. The environment involves three tier architecture which separates data processing (applications or Web server) from interface (client) and data storage (database server). Particular emphasis is on: The implementation of the IFC v1.5 product model in an Object-Oriented database to enable construction data to be represented, shared between applications and exchanged between the various partners involved in the project.

The Web server extensions are used to enable applications to be distributed and manipulated by users through Web browsers and other commercial software such as Excel, MS Project, etc. Each application can accept an HyperText Transmission Protocol (HTTP) request from a client, process it, and return a response to the client. Currently, the prototype supports the following applications: detailed design (exchanged through Part 21), specification, construction planning, estimating, data sheets, virtual reality and Drawing Web Format (DWF) representation of the design. The steps used in the development of such applications are described later in this paper. For the list of abbreviations, please refer to Appendix A.

2. Approaches to integrated environments

There have already been extensive studies in the area of computer integrated construction. Projects such as ATLAS [18], COMBINE [10,28], RATAS [11], ICON [9], COMBI [25], OSCON [8], OPIS [16], SPACE [4,1] and ToCEE [7] are a few examples of good work in this area. The work presented here builds on these previous projects and proposes an improved architecture for computer integrated construction using IFC, Web and CORBA standards [21]. The CORBA standard has been specified by projects such as COMMIT [12] and work at CSTB [33]. IFC is being developed by the International Alliance for Interoperability (IAI) [19].

Numerous studies have been carried out with the aim of integrating the various project life-cycle phases through IT solutions [9,16,4,29,30,32]. These studies have all proposed similar solutions involving bringing together the various design and construction functions such as project design, estimating, and construction planning into a single ‘computer’ environment, i.e., the development of an integrated construction environment. However, the principles on which the development of these integrated environments are mainly based on direct translators or neutral formats, blackboard architecture, and integrated databases, differ considerably.

The direct translators approach entails taking the input data from one system and converts it so that it can be used in a second system to represent the same information if possible. The blackboard approach again uses a third party medium (a blackboard) for the transfer of information where expandability and maintainability of managing the blackboard is an issue that need to be considered in the development process. The integrated databases approach focuses on the integration of construction applications through a product model, where information is stored and can be accessed by all the applications.

3. WISPER: an overview

3.1. Three-tier client–server architecture

As shown in Fig. 1, the architecture of WISPER uses a three-tier client–server infrastructure to

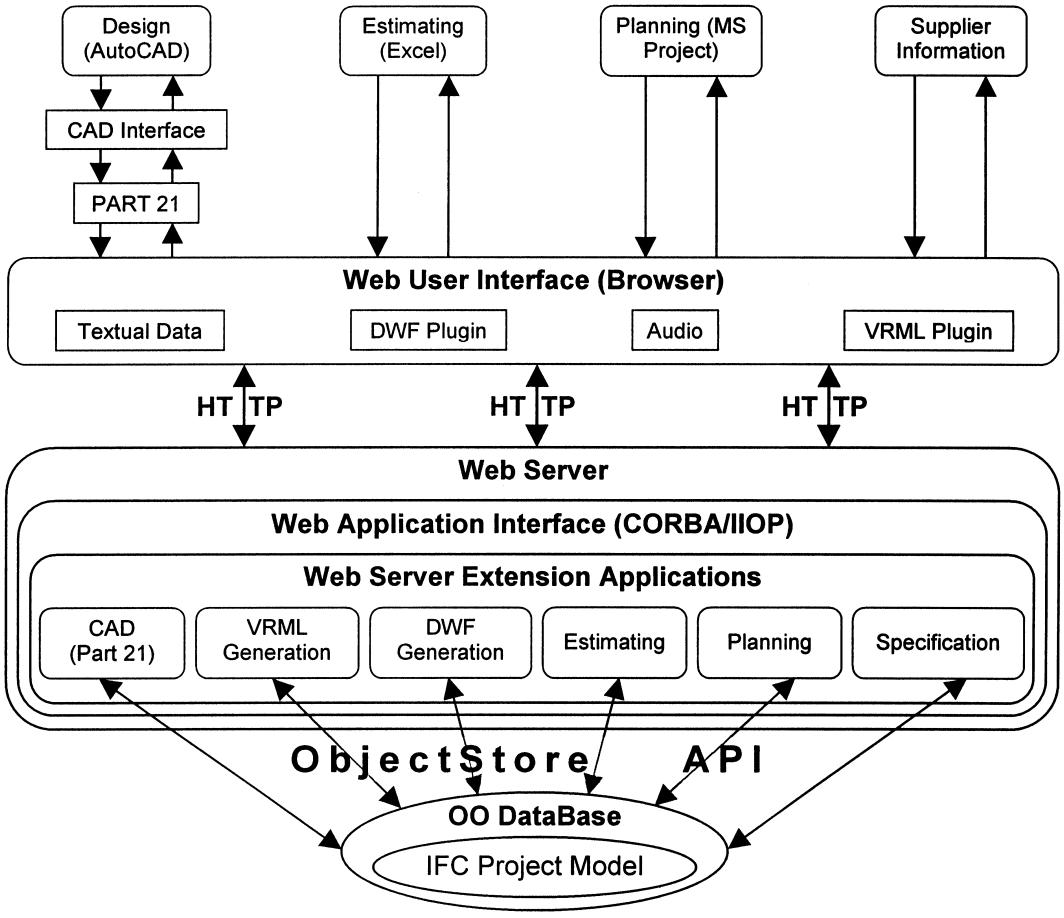


Fig. 1. Overview of WISPER.

demonstrate the integration between detailed design, building element based cost estimating, and construction scheduling, in addition to a VRML viewer that allows the graphical querying of a project database [3,5,14,15]. Within a three-tier architecture, each tier/facet of the application, i.e., presentation (user interface), logic (processing), and data storage (database), is isolated providing benefits such as maximum control, scalability, and flexibility [13,22,23].

3.2. IFC object-oriented project database

At the lowest level of the system architecture (Bottom Tier), the main project database has been developed using the ObjectStore object-oriented database management system together with the implementation of Java classes derived directly from the Final Version 1.5 of IFC. The IFCs are an initiative by leading construction firms and software vendors and to produce a standard data model for interoperability of construction related software applications [19]. The project model for version 1.5 of the IFCs has been published using EXPRESS (ISO10303-11). EXPRESS is a formal language developed by the Standard for the Exchange of Product Model Data (STEP) community for the description of data models. To map this schema to ObjectStore, ST-Developer was used to convert the IFC schema from EXPRESS to a set of programming language classes (pure Java classes). ST-Developer is a STEP Tools product that provides a comprehensive, stand-

alone set of software development tools for handling the complexities of the STEP standard, which can be used to build software that works with STEP data in object-oriented databases, relational databases, and traditional files [27]. ObjectStore utilities are then used to read the Java classes to create the IFC project database schema. This schema can then be used and instantiated by the various construction applications.

3.3. Web / application servers

Using Web client–server technology, i.e., Web Server Extensions, applications have been implemented as Java Web Application Interface (WAI) applications for the two-way exchange of information between the project database and existing software, i.e., AutoCAD, Excel, MS Project scheduling package, etc. (Middle Tier), WAI is a CORBA-based programming interface that defines object interfaces to the HTTP request/response data and server information. Via WAI, the Web applications accept an HTTP request from a client (Web browser), process it, and return a response to the client.

3.4. User interface

Finally, the top tier of the system provides the interface to the overall system. The user interface has been implemented as a set of Web pages and other commercial applications to support the different construction professionals that may need to interact with the environment. The user interface enables the user to submit, retrieve, process and manipulate data. Menus are created within each Web page. Users can perform operations on the project data by selecting the appropriate menu item from the Web page. Each menu item references a Uniform Resource Locator (URL) that submits an HTTP request to the server. The Web server assigns the responsibility for processing the HTTP request to the CORBA objects.

WISPER takes advantage of open Internet standards wherever possible and uses the concept of a 'project Web site' as a first point of access for all shared project information. Each new project has its own Web site, allowing project information to be accessed through an internal network (Intranet) or through the wider global Internet. Furthermore, existing security mechanisms such as password protection, certificates, IP address filtering and so on ensure the security of the project information by controlling access to the project Web pages.

4. Implementation of WISPER

The software tools that have been used in the development of WISPER are listed in Appendix B. However, prior to the implementation, the research team set out to investigate the requirements for such an environment from the end user point of view, i.e., the construction industry. The ''use case'' technique was selected which is the subject of discussion in Section 4.1.

4.1. Use cases

Use cases is a technique for eliciting, understanding and defining functional system requirement [20]. Developing software is normally a process of interaction between a procurer and a developer. It is vital that during this process both sides talk the same language for how the system to be developed should work or be used. The use cases explore the critical requirements which have to be met by the system. A number of use cases have been identified for the development of WISPER. Each use case is implemented by a corresponding work package. These use cases, detailed descriptions of which can be found in Ref. [2], are:

- Case 1 — Establishing a new project database.
- Case 2 — Populating a database with design information.
- Case 3 — Generating cost estimates.
- Case 4 — Generating specification of building elements.
- Case 5 — Linking specifications to individual building elements.
- Case 6 — Browsing data through a VRML viewer.
- Case 7 — Generating construction planning information.
- Case 8 — Accessing data sheets.

4.1.1. Example: case 3 — generating cost estimates

All the building elements that form the design and their number of occurrence will be made available to

the user for costing purposes. System interaction is determined as follows:

(a) The estimator accesses and logs into the project's web site using a standard web browser.
(b) The estimator navigates to the cost estimating web page where he/she is presented with a choice either to retrieve all or part of the list of the building elements together with their quantities.
(c) A spreadsheet application is launched to allow the user to add cost data to the building elements. The user will also be able to create cost items which are not directly derived from building element information e.g., preliminaries. The cost elements should be flagged.
(d) The estimator invokes a program to upload the cost data back to the project's server.

4.1.2. Example of work package: WP3 cost estimating based on building elements

Goal: To provide a mean of exchange bills of quantities between the project database and a spreadsheet for the purpose of adding cost information (see use case 3). The work will be carried out in three phases (of which only phase one is illustrated in this paper):

4.1.2.1. Phase 1

T1: Develop an application to retrieve building elements and quantities into a spreadsheet. This application will be a WAI server extension. When accessed through its URL, the application will generate a file containing a bill of quantities from the database and return it to the user for import into a spreadsheet. For purposes of demonstration Microsoft Excel will be used to access this data.

T2: Develop an application to write an updated cost estimate into the database. This application will be a WAI server extension. The application will update the project database when files are sent via an HTTP POST request.

T3: Develop a web based interface for the above. This will involve creating a web page containing the links needed to access the server application developed in task one.

T4: Develop an application to post costed building element back to the server. This application will use the HTTP POST protocol to upload the costed element data file from the local hard disk to the application developed for task two. The cost data will be exported from the Microsoft Excel.

4.2. WISPER user interface

Taking advantage of the widespread use of the Web through the browsers, all interfaces to WISPER's applications have been developed through the Web, i.e., they can be accessed using a commercial Web browser. A set of generic Web pages has been created to act as an interface to the overall integrated environment. These pages are used as a template for the creation of project Web sites. They contain further Web pages for accessing the various project applications and other project information such as actors and contact addresses. Fig. 2 shows the upload design function which uploads the design file to the database of the project specific Web page, which enables users to access the various construction applications by selecting the required application from the left hand frame of the web page. Application Web pages contain pointers (in a form of URL) which enable certain functions to be executed.

4.3. Applications

In a previous section, we have described how the database was created from the IFC schema. The schema is instantiated by the various construction applications such as design, planning and estimating. In this section the applications implemented in WISPER, which each support a subset of the project lifecycle, will be described.

4.3.1. Computer aided design (CAD) application

Due to the absence of commercial IFC compliant CAD systems, the research team implemented its own system that uses AutoCAD-14 geometry engine (AutoCAD 14 supports both 2-D drafting and solid modeling geometry). This application is capable of supporting both newly created and existing projects and is written in C++ (as AutoCAD does not support Java). The following sections explain this application through a number of stages.

4.3.1.1. Creating new design projects. The purpose of this application is to enable the user to create new building designs. The application maps the design

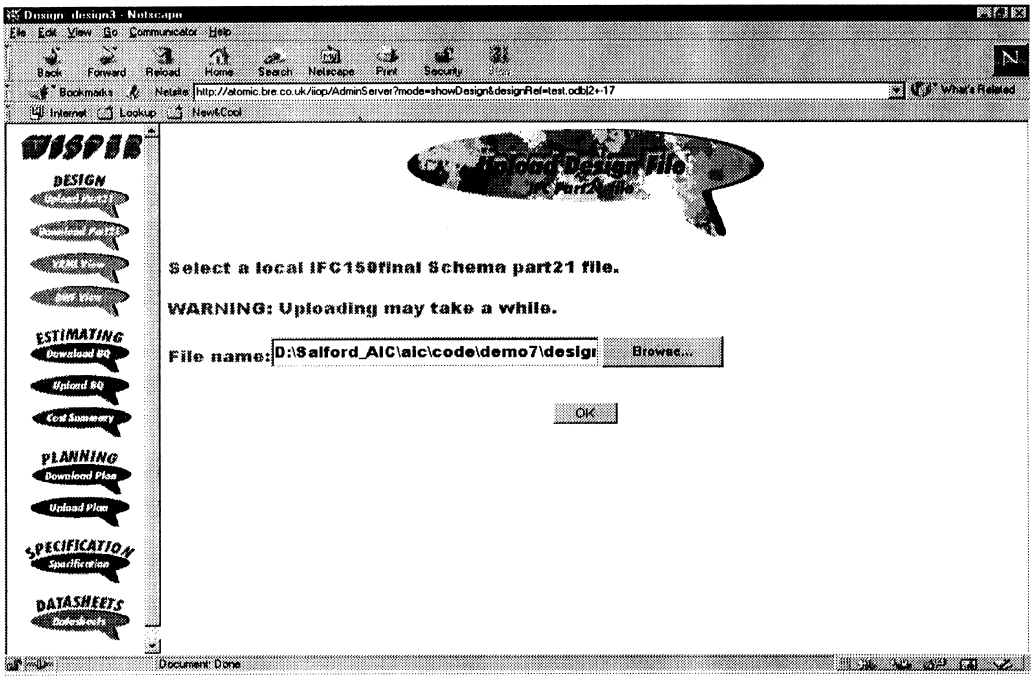


Fig. 2. Sample of a Web page from the user interface.

information from ''dwg'' format to the IFC and instantiates the IFC project model. Project data can be exchanged by writing the information to STEP Part 21 file or shared directly by all applications accessing the data in the project model using Object-Store API. The design information is saved as a solid model from which the 2-D wire-frame is derived.

A WISPER menu which contains all the supported building elements is added to the AutoCAD menu bar which enables the user to select a building element (e.g., wall) and position it in the desired location. In order to capture the data for the re-generation of the design at later sessions, the following will need to be determined:

- Object definition — The shape of the building element and how it is to be drawn by the CAD system is determined by extracting the nodes that form the profile of the building element from the Object-ARX of AutoCAD (see Fig. 3). In order to create the volumetric building element, the depth of the object is also extracted to describe the element as a solid geometry by extruding the profile along one of the building element's local coordinate axis for a distance equal to the depth of the object. The information then can be mapped to a format that can be read by a CAD system to view and manipulate the information, in this case AutoCAD.

Currently, WISPER supports all the building elements that are defined in v1.5 and does not have any tools that would enable the definition of building elements by dynamically grouping all the low level geometries of the building elements. It is expected

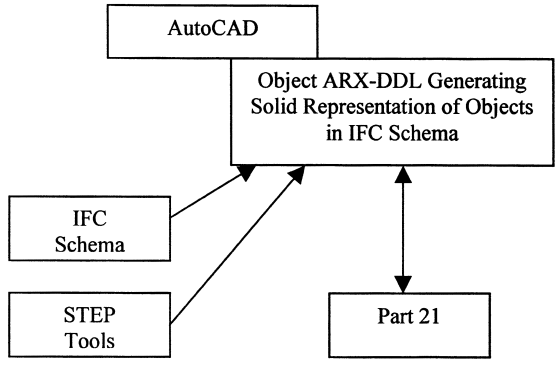


Fig. 3. Components of the CAD application.

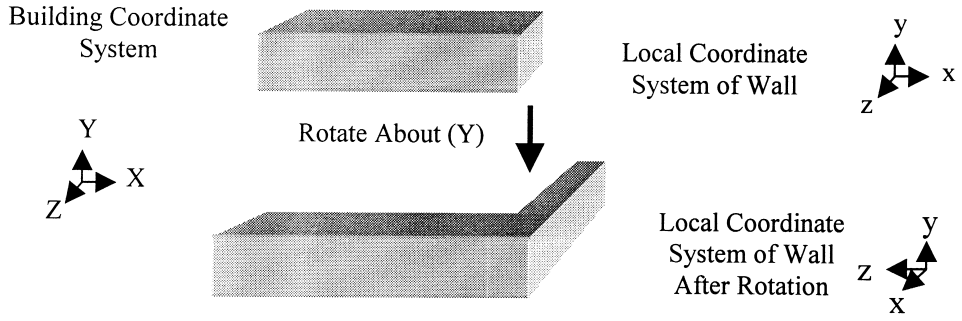


Fig. 4. Building element transformation.

that this system would be replaced by a future more sophisticated IFC compliant CAD system.

- Position — the position of the building element relative to the building, e.g., the position of the wall relative to building.
- Orientation — the orientation of the building element relative to the building coordinate axis, e.g., which axis of the building coordinate system corresponds with the *x*-axis of the local coordinate axis of the building element.

When building elements are defined, their initial position and orientation relative to the building coordinate system are known. As they are moved to the desired position in the building, they are likely to be rotated and transformed (see Fig. 4). As a result, the new position will need to be calculated. The position of a building element relative to the building is calculated by tracking the transformation of the building element when it is applied. Consider the wall shown in Fig. 4. Initially, its *x*-axis corresponds to the *X*-axis of the Building Coordinate System. Notice that after the wall has been applied to the building, its *x*-axis corresponds to the *Z*-axis of the Building Coordinate System. Hence, there is a need to calculate the rotation of each building element in order to establish its orientation relative to the building.

To represent openings such as doors and windows, regularised boolean operations will also need to be represented. An opening can be represented as the regularised difference between a volume that corresponds to the opening and the volumetric building element in which the opening is placed.

4.3.1.2. The IFC project model. Using the IFC schema and the ST-Developer tool, the data is written out to STEP Part 21 file which can be loaded to ObjectStore to instantiate the IFC schema. Fig. 5 shows the hierarchy of the building element in the IFC project model. The geometric information of each building element is not held in the *IfcBuildingElement* level, but at a higher level, i.e., the

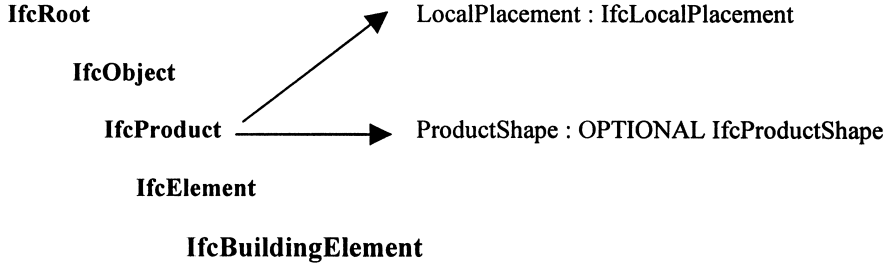


Fig. 5. Building element hierarchy.

IfcProduct. This enables the aggregation of product component shapes at the geometric level.

The *IfcProduct* has two nodes; LocalPlacement, which is represented by IfcLocalPlacement node and ProductShape, which is represented by *IfcProductShape* node. The CAD application writes the placement of each building element to the *IfcLocalPlacement* node. It is capable of describing the placement of building elements in 3-D space by storing the position and orientation of both 2-D (*Ifcaxis2placement*2d) and 3-D (*Ifcaxis2placement3d*) geometric elements. In 3-D, the placement of the element is defined in terms of a point and two orthogonal axes. *Ifcaxis2placement*3d contains the following attributes: Location, Axis and RefDirection. For example the *Ifcaxis2placement* for a solid wall is presented as follows:

- Location: IfcCartesianPoint — this describes the origin of the local coordinate system (LCS) of the building element ''wall'', the default of which is defined as the lower-bottom left hand corner.
- Axis: OPTIONAL IfcDirection — the exact direction of local *Z* axis
- RefDirection: OPTIONAL IfcDirection — the direction of the local X axis

The node *IfcProductShape* in the IFC model describes the object's shape. Each building element is defined as an extruded solid. Each building element could have one or more segments. The resulting solid is the union of all the segments. Each segment is described by a solid area defined by a profile, this area is then extruded along the extrusion direction. The extrusion is always in the direction of the swept area. The profile and extrusion path data of each element are described by an *IfcPolyLine*, which contains the points that define the profile. This data is then written to the node *IfcAttDrivenExtrudedSolid*. This enables different representation of objects to be generated (e.g., wireframe, solid representation, VRML).

4.3.1.3. Uploading and downloading design information to and from the IFC database. This stage populates the database from STEP Part 21 file and is implemented as a Netscape WAI Web server extension [24]. The user interacts with this stage through a Web page. The WAI application creates and registers CORBA objects with the Web server associated with project database. The Web server associates each of these objects with a specific URL of the form: http://server//iiop/object_name; where 'server' identify the name of the machine that the Web server runs, 'iiop' is Internet Inter ORB (Object Request Broker) Protocol which passes requests or services replies through the internet, and the 'object_name' is the name in which the application is registered and can be identified by the ORB. By accessing this URL, the Web server delegates the responsibility for processing the HTTP request to the CORBA objects. In this case, the application processes Part 21 files sent to it using the HTTP POST protocol. ST-Developer [27] uses the IFC schema and its corresponding Java classes to instantiate the Java classes that correspond to the project instances defined in Part 21. These can then be loaded to ObjectStore using ObjectStore utilities (see Fig. 6).

Similarly, Part 21 files of existing projects can be loaded to the database if amendments or modifications need to be carried out. The modified file can then be saved in Part 21 file format as shown in Fig. 7. This part of the application is written in Java.

4.3.2. Browsing a design using VRML

The purpose of this application is to generate the VRML representation of the design and display the

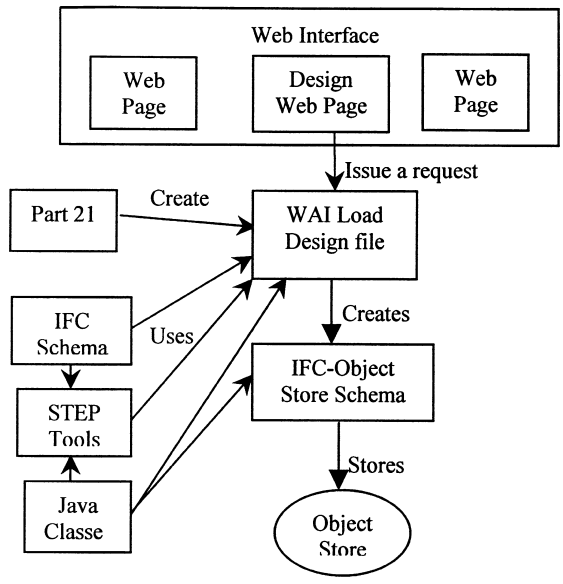


Fig. 6. Creating new design in the database.

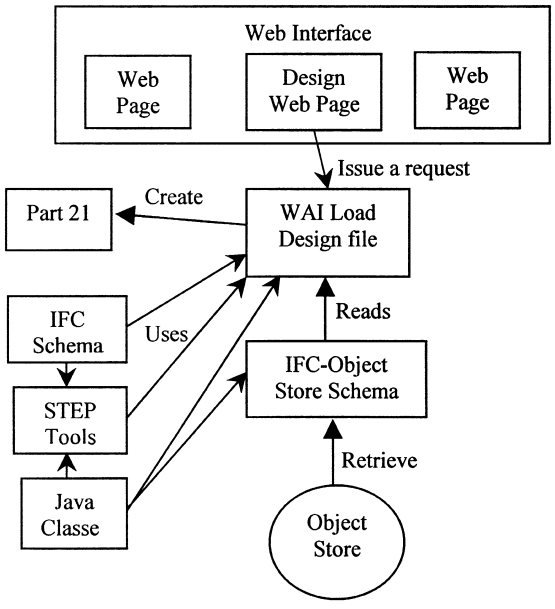


Fig. 7. Loading and editing existing design.

results within a Web browser. Users are able to interact with the VRML model and to retrieve other construction data which is associated with an object or the project as a whole. Therefore, the virtual reality model is not only used for visualization purposes but also as a means of data retrieval.

The application ''WAI Create VR Model'' is implemented as a WAI server extension application, as shown in Fig. 8, which is written in Java. Before the user can interact with the application, the design needs to be uploaded to the ObjectStore database and the virtual reality (VR) server application is invoked so that it can process requests sent to it by the client application. The user interacts with the application from the client tier through a Web page. When the VRML's URL is accessed from the Web page, the server processes the request and delegates the responsibility to the ''WAI Create VR Model'' application. The WAI application processes the request and responds accordingly by either displaying the VRML representation on the client tier, generating textured images or retrieving specific information about a particular part of the building from the database.

The ''Retrieve Project Data'' application (Fig. 8) parses the database to identify the building elements and their relationships with other elements and generates the VRML representation of the design. It also retrieves the information about any selected element in the VR environment. This is done through accessing the reference of the object concerned which has been allocated at the time of reading Part 21 file, i.e., inserting each read object with a unique reference into a hash table. This approach has enabled much faster retrieval of data and offers a better solution than the linear searching of objects.

4.3.2.1. Mapping IFC geometry to VRML. The approach that has been used to generate VRML representation is the extrusion of a 2-D profile along a 3-D curve. This approach has enabled developers to generate the VRML representation for a range of common building elements and other objects that might be found in a construction project. Fig. 9 shows a simple VRML file that describes a simple building element (rectangular floor) in order to understand the information required to generate the VR model automatically and in a manner which is totally transparent to the user.

The *Transform* node in VRML creates a new coordinate system for the building element relative to its parent (storey) coordinate system. *Shapes* are created as children of the Transform node and are built relative to the new coordinate system's origin.

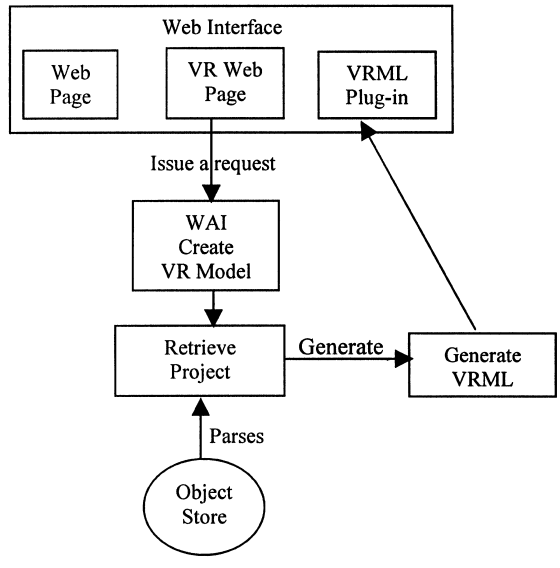


Fig. 8. Creating the VRML representation of the design.

```
#VRML V2.0 utf8
# Floor
Transform {
   rotation 0.0 1.0 -0.0 6.283185307179586
   translation 0.0 -300.0 -0.0
   children Shape {
     appearance Appearance { material Material { diffuseColor
          0.5 0.5 0.5 }}
     geometry Extrusion {
       solid FALSE
       crossSection [
                    1600.0 -3600.0, 8800.0 -3600.0, 8800.0 -9200.0,
                    1600.0 - 9200.0, 1600.0 -3600.0,
        ]
       spine [
       0.0 0.0 0.0,
       0.0 300.0 0.0
       ]
     }
   }
}
```

Fig. 9. VRML file — rectangular floor.

The *Extrusion* node contains two main fields; the *crossSection* and *spine*. The *crossSection* defines the 2-D profile of the shape to be extruded while the *spine* specifies a list of 3-D coordinates that defines the path along which the cross-section is swept to create the extrusion. As the cross-section sweeps along the spine, it leaves behind it a group of faces that create the skin of the extruded shape.

For every building element in the database, the application accesses the *IfcProductShape* of that element to obtain the geometrical information associated with the element. The points that form the profile of the object, depth of the object, position and rotation are retrieved from the IFC model which is defined in ObjectStore and populate the VRML's Transform node, crossSection and spine fields of the Extrusion node, respectively [6]. The retrieval process of data is the reverse of populating the IFC schema by the CAD application. This approach is valid for the majority of building elements. However, due to the lack of support to boolean operations in VRML, this approach would not work for elements that contain opening such as a wall with a window. To overcome this exception, walls with openings are created as a list of blocks that surround the openings such as the window and/or doors. For example, a wall with one opening will consist of four blocks grouped together as shown in Fig. 10.

4.3.2.2. Viewing the file on the Web browser. Once the virtual world is created by the application, the WAI application sends the information to the browser on the client tier with the Multipurpose Internet Mail Extensions (MIME) and this informs the Web Browser that the document to be sent is of VRML

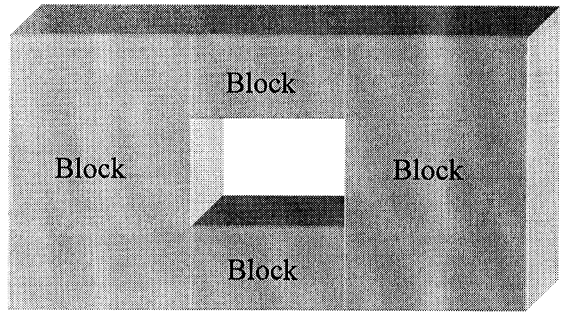


Fig. 10. Wall with opening created from blocks.

content type. MIME is a software standard that defines a mechanism to describe the type of content in a file sent via the Internet. All Web browsers understand MIME content type and use them to decide automatically how to display the information. Currently, VRML uses the x-world/x-vrml MIME content type. To view the VRML document, a VRML plug-in is required. In WISPER, the CosmoPlayer plug-in with Netscape Communicator has been used. Fig. 11 shows a VRML representation of a simple building displayed on a Netscape Communicator browser with CosmoPlayer plug-in, the bottom frame shows the retrieved information associated with the a door instance. This is generated by clicking on the building element. The retrieval process is described in section ''Browsing a Design Using VRML'' section.

The user interface enables the user to generate textured images or view a particular part of the building, for example all the building element on storey_1. This option can be used to support different views of the project information such as displaying all the structural elements found on a particular storey.

4.3.3. Estimating

The purpose of this application is to enable estimators to retrieve the relevant information directly from the design and to be able to make their estimates available to other parties. A Web server extension application (WAI) has been developed for this application (see Fig. 12). It initially generates the elemental IFC cost group objects, within the central project database, from the design. Elements are grouped together based upon their type. The application then calculates the elements' total quantities before returning this information into an Excel spreadsheet. Users can cost the cost groups in the Excel spreadsheet, save the Excel spreadsheet in a Comma Separated Value (CSV) format and upload

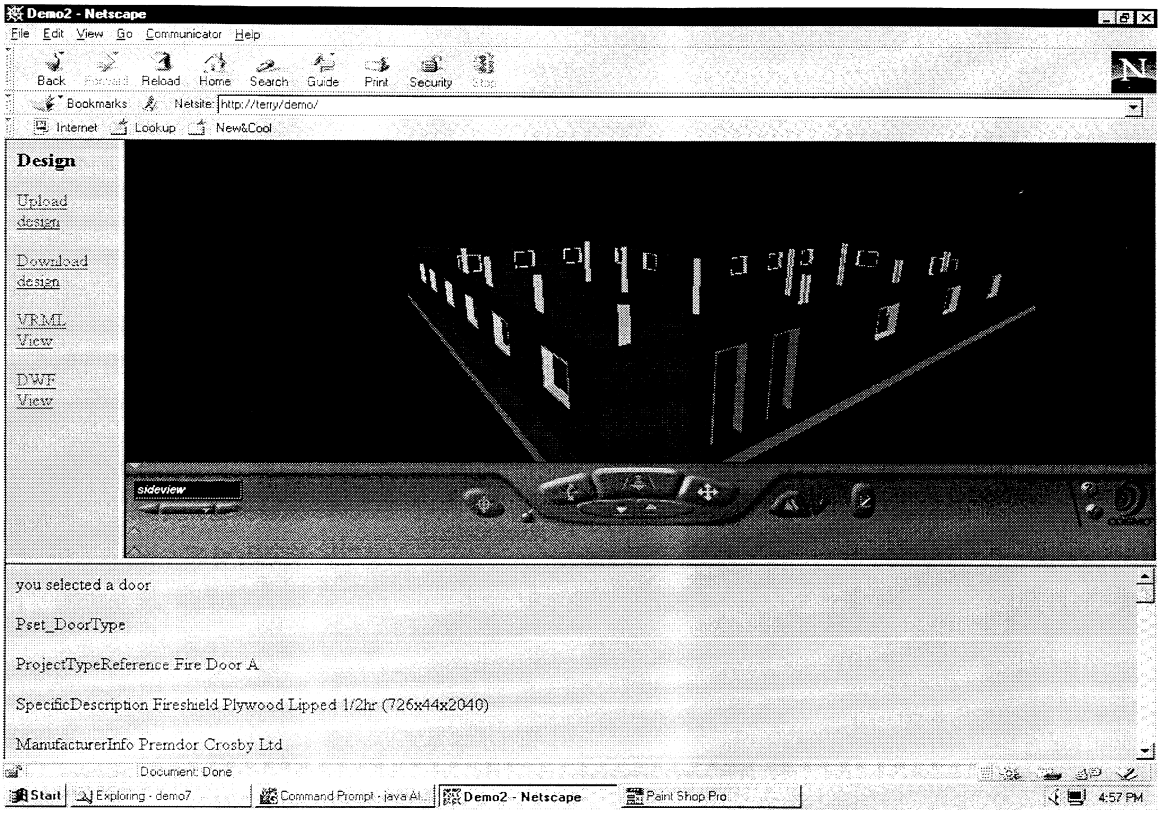


Fig. 11. A simple building generated with VRML.

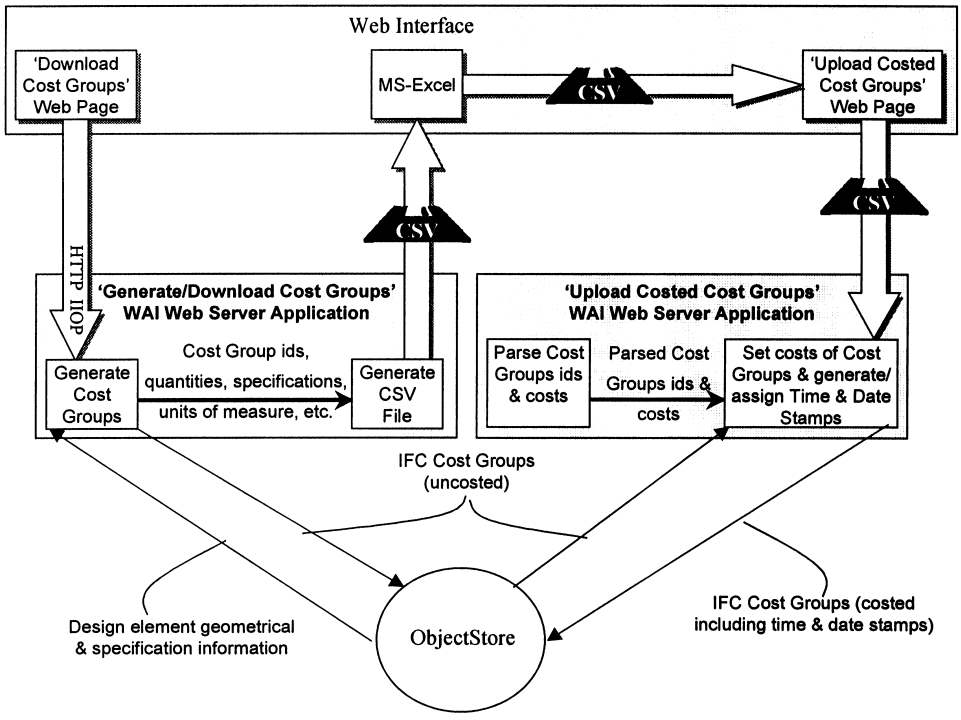


Fig. 12. Generating cost estimate.

the CSV file into the project database. This part of the process adds costs, date and time to the previously generated cost group objects. Finally, users can view a cost summary of the project in terms of both elements' type and storey.

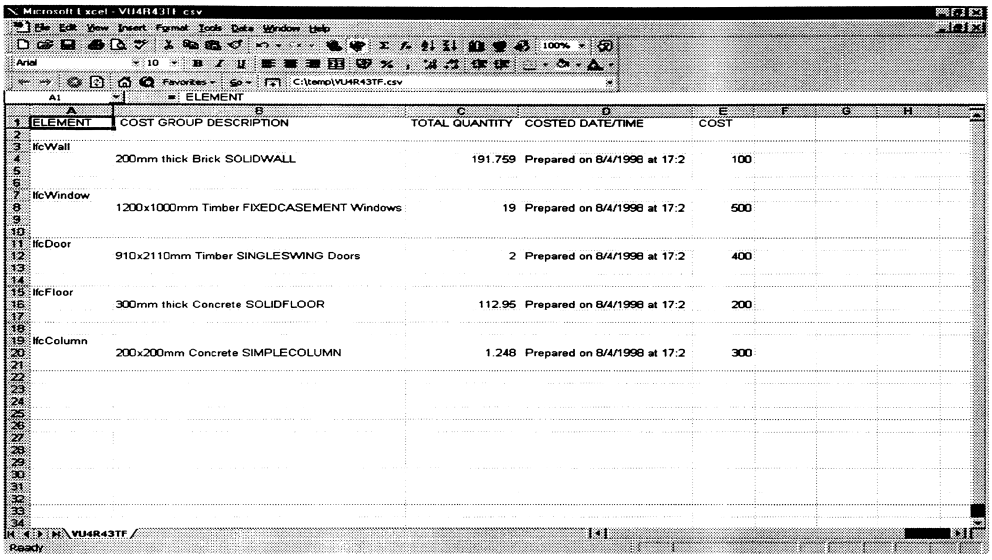
Once the user requests the download of the estimating information via the appropriate Web page, the WAI application initially searches the project database and identifies groups of elements based upon size, material, and generic type, e.g., 200 mm thick Brick SOLIDWALL, 1200 × 1000 mm Timber FIXEDCASEMENT Windows, etc. Next, for each group identified, two objects are created, i.e., an *Ifccostelementgroup* and an *Ifcrelcostscheduleelement*. The *Ifccostelementgroup* defines a group within a cost schedule, while the *Ifcrelcostscheduleelement* defines the relationship between a cost element group and those elements associated with it. Once these two objects have been created, their attributes are subsequently populated. In the case of the *Ifccostelementgroup* object, the Grouptitle attribute is set to the derived group description, e.g., 200 mm thick Brick SOLIDWALL, etc. From the point of view of the *Ifcrelcostscheduleelement* object, three attributes are populated as follows:

(i) The total group quantity is calculated from the elements' geometry in the case of wall, columns, beams, slabs, etc., and per item in the case of doors and windows, which in turn is set as the value of the Quantity attribute.

(ii) The Relatedobjects attribute is set as a list/array of the elements associated with the cost group.

(iii) The Relatingobject attribute is set to the relating *Ifccostelementgroup* object.

The cost groups are returned in a CSV file that is dynamically opened in Excel on the Client machine. As shown in Fig. 13, each cost group is returned with its cost group description (id), presented in terms of element type, i.e., Ifcwall, Ifccolumn, etc., and the total quantity. The user can cost each group and save the CSV file to their local hard drive before



ELEMENT	COST GROUP DESCRIPTION	TOTAL QUANTITY	COSTED DATE/TIME	COST
IfcWall	200mm thick Brick SOLIDWALL	191.759	Prepared on 8/4/1998 at 17:2	100
IfcWindow	1200x1000mm Timber FIXEDCASEMENT Windows	19	Prepared on 8/4/1998 at 17:2	500
IfcDoor	910x2110mm Timber SINGLESWING Doors	2	Prepared on 8/4/1998 at 17:2	400
IfcFloor	300mm thick Concrete SOLIDFLOOR	112.95	Prepared on 8/4/1998 at 17:2	200
IfcColumn	200x200mm Concrete SIMPLECOLUMN	1.248	Prepared on 8/4/1998 at 17:2	300

Fig. 13. Cost groups dynamically downloaded into Excel.

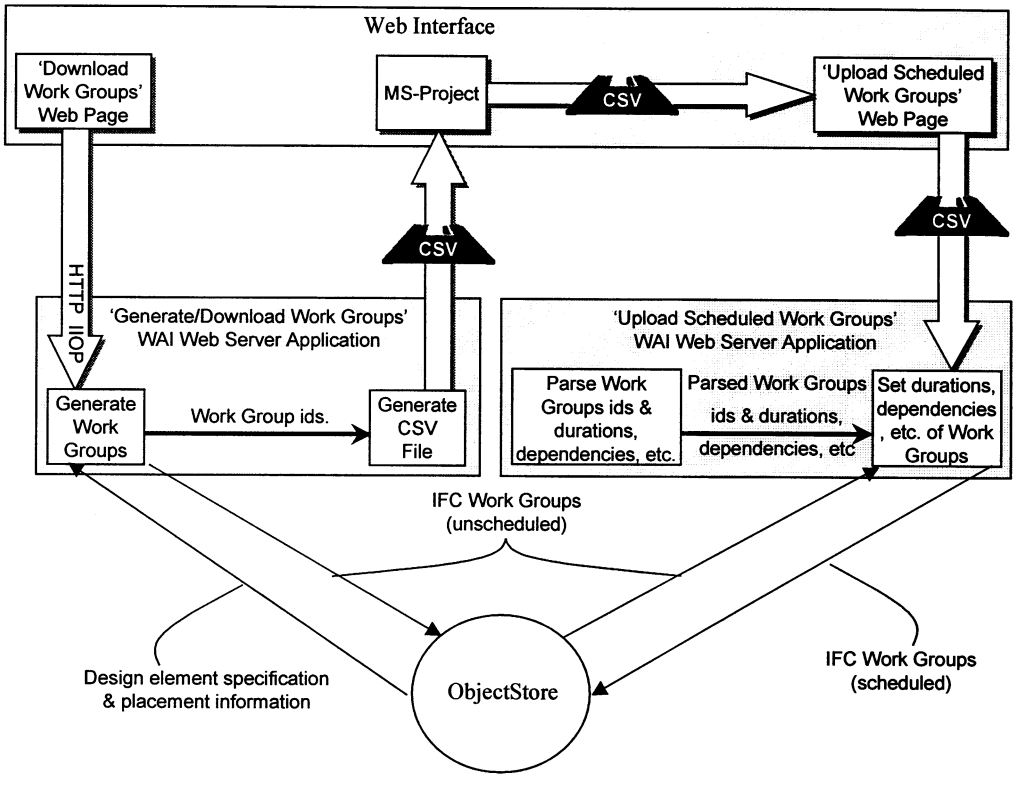


Fig. 14. Generating planning tasks.

uploading the file to the central project database via the appropriate Web page. When this information is loaded again in Excel, two additional columns are created, i.e., 'COSTED DATE/TIME' and 'COST', to include information on cost and when the cost was prepared. Detailed information regarding this process can be found in Ref. [31].

In addition, once the costs have been uploaded, the user (such as the project manager, client, etc.) has the option to view a cost summary of the project. Once requested, the application returns the cost summary within the Web browser (HyperText Markup Language, HTML) summarising the project costs by both storey and elemental type.

4.3.4. Planning

The purpose of this application is to enable planners to retrieve the planning information directly from the design and to make it available to other parties. The Web server extension application (WAI) which has been developed for planning initially generates the work group objects within the central project database from the design, before returning this information into an MS Project scheduling package (see Fig. 14). Users can then complete the construction plan by adding duration and activities' dependencies. The MS Project plan is saved to CSV format and uploaded into the project database. This process adds the duration and relationships to the previously generated work group objects.

Once the upload of the work groups has been requested from the appropriate Web page, the application commences by identifying each group of elements, i.e., Ifcbeam, Ifcwall, etc., at each storey from the building storey container object's (*Ifcrelcontains*) Relatedobjects attribute. For each group of elements at a particular storey, a work group is created, e.g., Construct Walls At Storey 0. A work group comprises of three objects, i.e., *Ifcworktask*, *Ifcworkschedule*, and *Ifcrelusesproducts*. The *Ifcworktask* object represents an identifiable unit of work to be carried out independently of any other units of work. The description of the task to be undertaken, as it will appear in the construction plan, is set as the value of the object's Taskdescription attribute. The *Ifcworkschedule* object allows for the allocation of date and duration to the work task, whose attributes are set from the upload of a construction plan. The *Ifcrelusesproducts* 'relationship' object handles the assignment of products as input or output of processes. Therefore, the Relatingobject attribute references the *Ifcworktask* object while the

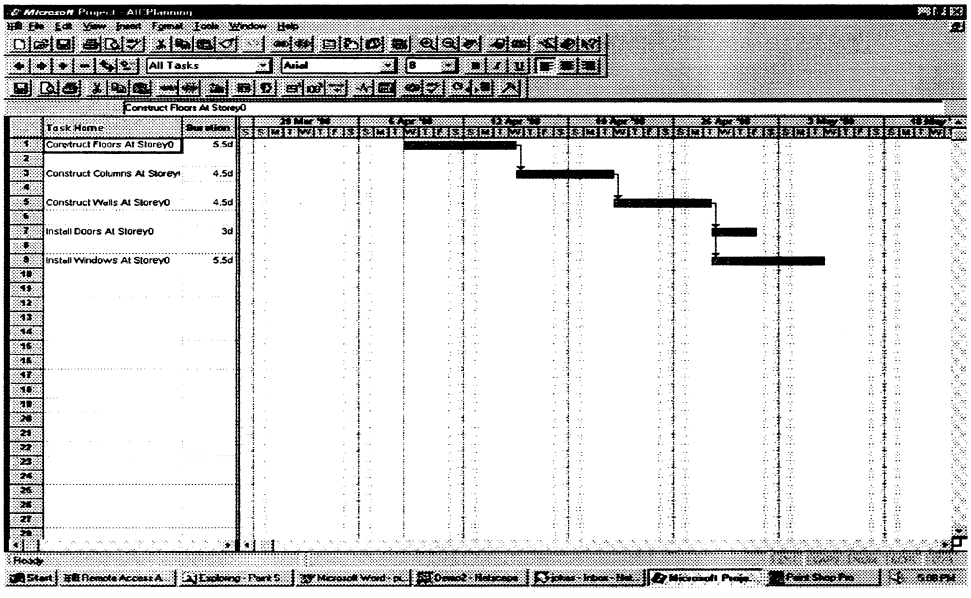


Fig. 15. Work groups dynamically downloaded into MS Project.

Relatedobjects attribute references the products/building elements resulting from the output of the task, i.e., Ifcwall, Ifcdoor, etc. Finally, the work groups are returned in a CSV file that is dynamically opened in MS Project on the Client machine (Fig. 15).

With the work groups opened within MS Project, the user is able to add duration together with the necessary dependencies and save the CSV file to their hard drive, before uploading the file back into the central project database. Once the upload request has been made, the Web server extension application parses the CSV file to identify the scheduled start and finish dates, duration and dependencies associated with each work group. For each work group parsed, an *Ifccalendardate* object is created for both the scheduled start and finish dates. Next, the Daycomponent, Monthcomponent, and Yearcomponent attributes of each object are set to the respective values, the Scheduledstart and Scheduledfinish attributes of the work group's associated with the *Ifcworkschedule* object are set to reference the *Ifccalendardate* object and the *Ifcworkschedule*'s Scheduledduration is set. Finally, where dependencies exist between work groups, an Ifcrelsequence object is created to handle the sequence relationship between two processes. The Relatingobject attribute is set to reference the predecessor process (*Ifcworktask*) and the Relatedobject attribute is set to reference the successor process. Following the uploading of the CSV file into the central project database, the work groups, together with their scheduled start and finish dates, duration, and dependencies, can be downloaded and returned into MS Project as shown in Fig. 13.

Currently, the Web server applications for both estimating and planning generate the information in CSV format. It is possible to amend the server applications to return the information in HTML format. Further, Microsoft Internet Explorer v4.0 does imbed MS Project and Excel, while Netscape Communicator externally spawns both Ms Project and Excel.

4.3.5. Specification and suppliers information

This application enables the user to define dynamically the design element specifications for a project via a product supplier database Web-site. The application has been implemented as two WAI applications. The first of these applications enables the user

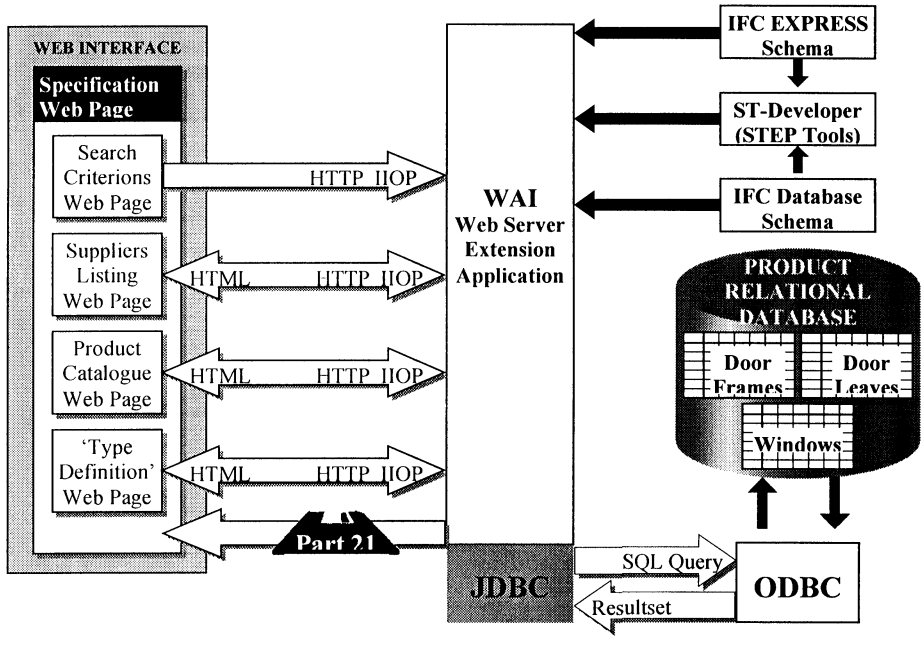


Fig. 16. Overview of the specification application.

to define the necessary door and window specifications from the supplier's database, which are returned to the user as IFC type definitions, shared property sets, etc., in the form of a STEP Part 21 file. Once the Part 21 file has been returned, the user can upload this file into the IFC object oriented project database, via the second of the WAI applications. The IFC v1.5 only defines the property sets for two building elements, i.e., doors and windows.

The application receives an initial 'search' request from the search criterions web page of the specific product, i.e., single swing doors, fixed casement windows, etc., via the HTTP POST protocol (Fig. 16). When a particular manufacturer/supplier is not specified within the request, the application connects to the product database via JDBC and ODBC, and retrieves those manufacturers/suppliers that manufacture/supply products matching the users' search criterions. Via the ODBC API, JDBC enables Java programs to establish a connection with a relational database, send SQL statements, and process the results. A HTML page presenting the manufacturers/suppliers received from the SQL query and incorporating links for viewing their products is dynamically generated and returned to the user (Fig. 17). By selecting the 'View Products' link of a particular supplier from the web page, a further request is sent to the application using the HTTP GET protocol. Upon receiving this request, the application connects to the products database, and retrieves information such as product description, image file name, etc., for the products that match the initial search criterions. The application uses this information to dynamically generate a catalogue HTML page of the specific products, which subsequently returns the page to the user. However, where a manufacturer/supplier is supplied within the initial 'search' request, then the application simply connects to the products database, retrieves the necessary information for the products that match the search criterions, and generates a dynamic catalogue HTML page of the specific products, before returning the page to the user.

In the case of windows, a catalogue of window units is returned to the user. However, from the point of view of doors, the application initially returns a catalogue of door leaves. When a user selects a particular element in the catalogue HTML page, the element's description is copied into the respective text input form element on the definition Web page, i.e., 'Window' or 'Door Leaf' input box. Following

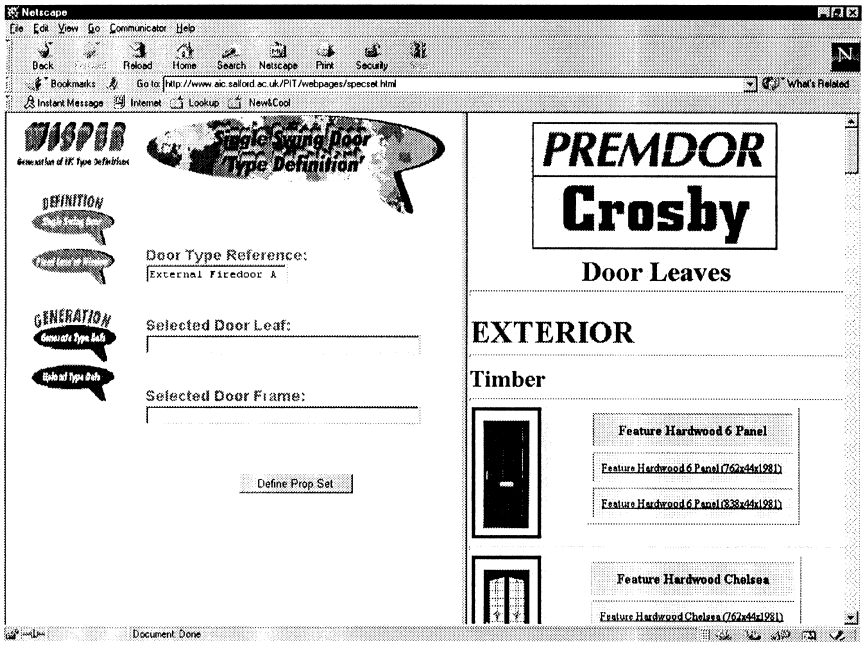


Fig. 17. Door definition and catalogue Web.

the selection of a door leaf, and the subsequent copying of its description into the 'Door Leaf' text box, a request is forwarded to the application to return a catalogue HTML page of door frames suitable for the door leaf selected. Using the door description passed in the request, the application retrieves the descriptions of the suitable door frames from the database, dynamically generates the catalogue HTML page and returns the page. When the user selects a door frame from the catalogue Web page, the description of the door frame is copied into the 'Door Frame' text input form element of the single swing door type definition Web page. Within the type definition Web page, the user specifies the element type reference in the 'Type Reference' input box, i.e., the reference ID property of the 'Pset_DoorType'/'Pset_WindowType' shared property set. Once the user has completed defining a type definition/specification, i.e., specified the element type's reference and selected the element/components associated with the element type, a request is sent to the application via the 'Define Definition' submit button on the type definition Web page. The application responds to the request by holding the type reference and the associated element/component description(s) passed in the request, which provides the opportunity for additional type definitions to be defined.

Following the definition of the required door and window type definitions, a final request is passed to the application to generate and return the type definitions previously specified in Part 21 exchange file format, i.e., an EXPRESS-driven data exchange file specification. The application acts upon this request by connecting to the product database and retrieving the various properties associated with the element(s) \component(s) specified for each type definition, e.g., dimensions, supplier, fire rating, material, etc. The complete IFC Type Definitions are generated and written out to a Part 21 physical file, which is returned to the user. This approach demonstrates the power of providing information over the Web. However, it requires product suppliers to tag the requisite information in their catalogues.

The second WAI application resides on the project database Web server. By sending the Part 21 physical file of the previously defined type definitions, via the relevant Web page, the application

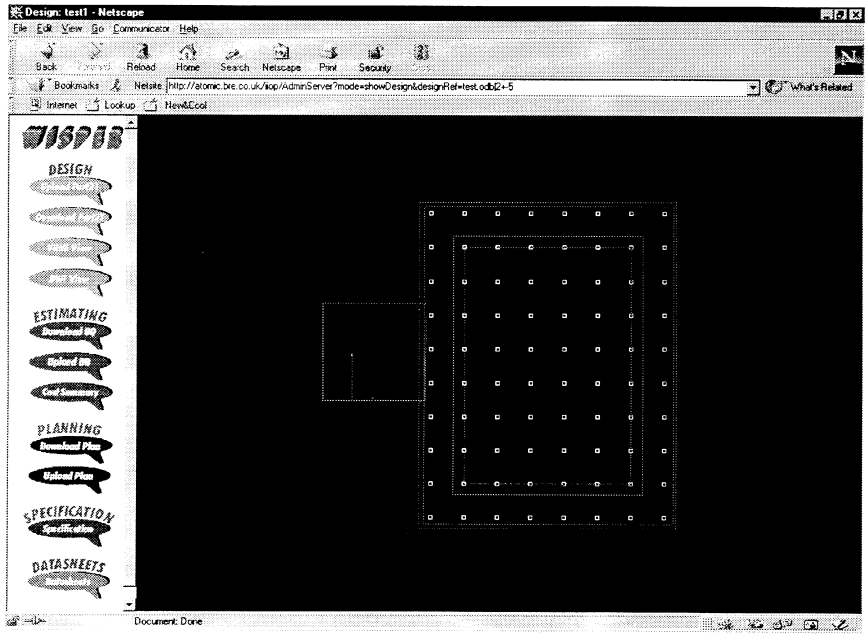


Fig. 18. The generation of DWF format.

delegates the responsibility of uploading the file to the IFC project database. Java objects are created that correspond to the project instances defined in the Part 21 file. Once uploaded to the IFC project database, the CAD application can download the Type Definitions and attach/link them to the relevant design elements as they are being designed.

4.3.6. DWF

A Web server extension application (WAI) has been developed for the generation of DWF from the IFC database, i.e., the design does not have to be in a specific CAD vendor format such as DWG. As a result, the DWF representation of any design created by any IFC compliant CAD system can be generated. DWF is a graphics format for the transfer of drawings over Intranets and the Internet. The drawing can be viewed using Web browsers with the WHIP plug-in. The application parses the database for the building elements that exist in a project and generate the DWF representation of each element. The generated information is then saved to a file and can be displayed within the WHIP plug-in (see Fig. 18). Users are able to interact with the elements of the 2-D design and to retrieve the relevant information.

5. Use of the environment

It is perceived that project managers could be a focal point in using the distributed computer integrated environment and would be responsible for ensuring smooth progress of work across the project's stages. Project information is carefully monitored and controlled by the project management organisation where it applies its own experience and rules to run the project effectively. The latter implies that such organisations might have their own databases and business logic which need to be accommodated in any integrated solution. This role can be facilitated by the introduction of the project management site (Fig. 19). The site can be based at and controlled by the project management team although it behaves as any other site in that it stores and exchanges project information with other sites. It acts as a repository for project information where:

- Multiple data versions of each discipline are automatically stored as and when they are released by their owning sites.
- Authorised practitioners from any discipline can access the data as and when required.
- Each transaction between the site and others is logged.

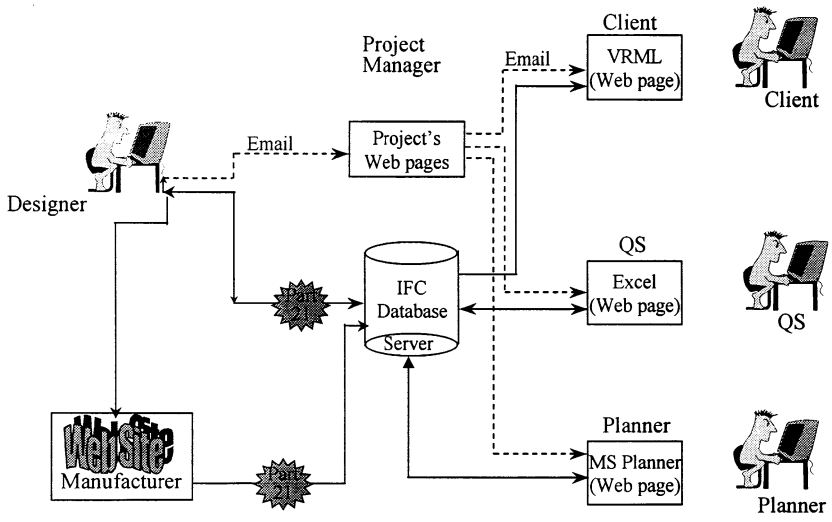


Fig. 19. The use of a distributed computer integrated environment.

- Other related/authorised sites are automatically informed of any new forthcoming versions of data.
- Communication between the project management site and other sites is carried out through the Web server.

6. Conclusion

This paper reports on the WISPER project which investigated the implementation of a new generation of computer integrated environments. The environment is based on a three tier architecture, where user interfaces, business logic and database are kept separate. It also investigated the implementation of the IFC project model as a core for such an environment. Web technology was used to develop the user interfaces and enable the communication of distributed applications.

A Web and IFC-based distributed computer integrated environment has been developed. This environment supports design (CAD), visualisation (VR and DWF), estimating, planning, specifications and supplier information. WISPER enables users to exchange project information through STEP Part 21 file and/or share the information through the IFC database. All applications can be accessed from Web pages. This gives greater flexibility and portability to all construction professionals to perform, monitor and manage their activities whenever and wherever they are. A full working version of the prototype can be found on http://www.aic.salford.ac.uk/Pit/.

This project is a consequence of two large projects which have been carried out at the University of Salford since 1993; SPACE and OSCON. The experience from these projects has significantly contributed to the development of this project. Moreover, the industry, which has led this project, has directed and contributed significantly to the various stages of its development. WISPER is considered a way forward towards achieving flexible and 'open' integrated computer environments for the construction industry. However, the technology by itself will not lead to full implementation of such environments. They need to be supported by a process-based model, a procurement system, and a design team with clear roles and responsibilities. The effect of applying a collaborative environment in the construction domain and its effect on the cultural issues has been investigated by Fruchter and Schmitt [17,26].

Acknowledgements

The authors would like to acknowledge the DETR for their research grant no 39/3/366 (cc1109) and John Laing, WS Atkins, Mathew Hall, and Autodesk for their valuable contribution.

Appendix A

CAD	Computer Aided Design
CORBA	Common Object Request Broker Architecture
CSV	Comma Separated Values
DWF	Drawing Web format
HTML	HyperText Markup Language
HTTP	HyperText Transmission Protocol
IAI	International Alliance for Interoperability
IFC	Industry Foundation Classes
ISO	International Organization for Standardization
LCS	Local Coordinate System
MIME	Multipurpose Internet Mail Extensions
ORB	Object Request Broker
STEP	Standard for the Exchange of Product Model Data
URL	Uniform Resource Locator
VR	Virtual Reality
VRML	Virtual Reality Modeling Language
WAI	Web Application Interface
WISPER	Web-based IFC Shared Project EnviRonment

Appendix B. Software components used in the development of the environment

- Microsoft Developer Studio 97: This integrated development suite is the most widely used Windows programming environment. It has compilers for both Java and C++.
- AutoCAD 14.0: AutoCAD is one of the most used CAD systems in the construction industry and has a number of add-ons.

- ST-Developer — STEP Tools: ST-Developer is a set of software tools for working with EXPRESS information models and EXPRESS data sets. It automatically generates code from EXPRESS.
- JDK 1.1.x — JavaSoft: JDK 1.1.x is the reference Java implementation and is supported by the ObjectStore OODBMS.
- ObjectStore v5.0 — Object Design: ObjectStore is a fully distributed object database. It has a recovery/restart and transaction management for concurrent access by multiple users. It also provides support for both Java and C++ programming languages.
- Netscape Communicator 4.x with SGI CosmoPlayer VRML and WHIP plug-in: The Netscape browser is used for all development. However, the final system is browser independent. The browser ships with a Java ORB which may be of some use for certain tasks, e.g., synchronizing the VRML viewer.
- Netscape Enterprise Server 3.x: Enterprise Server supports extension applications through CORBA objects that are language independent. This fits well with the idea of using a universal browser client to access all forms of project data. The built-in Java ORB is used in implementing some of the system functionality.
- Excel: Spreadsheet for displaying cost information.
- Microsoft Project v 4.1: Construction planning package for displaying construction tasks and their durations.

References

[1] AIC Research Group, SPACE: An Awareness Interactive Multimedia CD-ROM, BuHu Research Centre, University of Salford, UK, http://www.salford.ac.uk/survey/aic, 1996.
[2] AIC Group, Partner in Technology Project DOE Ref: 39/3/366 (cc1109), Software specification, October, Internal Report, Department of Surveying, Salford University, UK, 1997.
[3] AIC Group, Partner in Technology Project DOE Ref: 39/3/366 (cc1109), Final report, May, Internal Report, Department of Surveying, Salford University, UK, 1998.
[4] M. Alshawi, C.W. Putra, I. Faraj, A structured concept for objects life cycle in integrated environments, Microcomputers in Civil Engineering 12 (1997) 339-351, Blackwell.
[5] M. Alshawi, G. Aouad, I. Faraj, T. Child, J. Underwood, The implementation of the industry foundation classes in integrated environments, CIB W78 Conference, Sweden, August, 1998, pp. 55–66.
[6] A. Ames et al., VRML 2.0 Source Book, ISBN: 0471165077, 1997.
[7] R.W. Amor, M. Clift, R. Scherer, P. Katranuschkov, Z. Turk, M. Hannus, A framework for concurrent engineering — ToCEE, European Conference on Product Data Technology, PDT Days 1997, CICA, Sophia Antipolis, France, 15–16 April, 1997, pp. 15–22.
[8] G. Aouad, Integration: from a modelling dream into an implementation reality, Proceedings of CIB W55, International support for building economics, Lake District, UK, 1997, pp. 7–29.
[9] G. Aouad, M. Betts, P. Brandon, F. Brown, T. Child, G. Cooper, S. Ford, J. Kirkham, R. Oxman, M. Sarshar, B. Young, ICON (Integration of Construction Information): integrated databases for the design, procurement and management of construction, Final Report, Department of Surveying and Information Technology Institute, University of Salford, 1994.
[10] G. Augenbroe, An overview of the COMBINE project, proceedings, in: Scherer (Ed.), ECPPM'94: Product and Process Modelling in the Building Industry, Balkema, 1995, pp. 547–554.
[11] B.C. Björk, The RATAS project — an example of co-operation between industry and research toward computer integrated construction, Journal of Computing in Civil Engineering, ASCE 8 (4) (1994) 401–419.
[12] Brown et al., Promoting computer integrated construction through the use of distribution technology, Electronic Journal of Information Technology in Construction, 1996.
[13] R. Campbell, It all ends in Tiers, SIGC Application Development Advisor, November, 1997.
[14] I. Faraj, M. Alshawi, G. Aouad, T. Child, J. Underwood, Distributed object environment: using international standards for data exchange, to be published in Journal of Computer Aided Civil and Infrastructure Engineering (1999) Blackwell.
[15] I. Faraj, M. Alshawi, G. Aouad, T. Child, J. Underwood, The implementation of the IFC in a distributed computer integrated environment, Second European Conference on Product and Process Modelling in the Building Industry, London, UK, October 19–21, 1998, pp. 187–198.
[16] T. Froese, B.C. Paulson Jr., OPIS: an object model-based project information system, Microcomputers in Civil Engineering 9 (1) (1994) 13–28.
[17] R. Fruchter, Internet Web-mediated collaborative design and learning environment, Artificial Intelligence in Structural Engineering, 26–31 July, Ascona, Switzerland, Springer, 1998, pp. 384–397.
[18] R. Greening, M. Edwards, ATLAS implementation scenario, proceedings, in: Scherer (Ed.), ECPPM'94: Product and Process Modelling in the Building Industry, 1995, pp. 467–72.
[19] IAI, International Alliance for Interoperability, 1999, http://www.interoperability.com/.
[20] I. Jacobson, Use cases in large-scale systems, a Web document, 1995, http://www.unantes.univ-nantes.fr/usecase/root.html.

[21] OMG, OMG: Object Management Group, 1999, http://www.omg.org/.
[22] R. Orfali, D. Harkey, J. Edwards, CORBA, Java, and the Object Web, BYTE 2 (10) (1997) 95–100.
[23] D. Pountain, J. Montgomery, Web components, BYTE 22 (8) (1997) 56–68.
[24] Netscape Communication, CORBA: catching the next wave, 1997, http://www.netscape.com.
[25] R. Scherer, EU-project COMBI — Objectives and overview, proceedings, in: Scherer (Ed.), ECPPM'94: Product and Process Modelling in the Building Industry, Balkema, 1995, pp. 503–510.
[26] G. Schmitt, A new collaborative design environment for engineers and architects, Artificial Intelligence in Structural Engineering, 26–31 July, Ascona, Switzerland, Springer, 1998, pp. 384–397.
[27] STEP Tools, The ISO STEP Standards, http://www.steptools.com/library/standard/.
[28] M. Sun, F. Parand, Integration of CAD, product model and distributed building component database, Second European Conference on Product and Process Modelling in the Building Industry, London, UK, October 19–21, 1998, pp. 487–494.
[29] M.G. Syal, M.K. Parfitt, J.H. Willenbrock, Computer-integrated design/drafting, cost estimating and construction scheduling, Housing Research Center Series Report No. 11, The Pennsylvania State University, Department of Civil Engineering, University Park, 1991.
[30] A. Tracey, T. Child, G. Aoud, P. Brandon, Y. Rezgui, Developing integrated applications for construction: the OSCON approach, First International Conference on Computing and Information Technology for Architecture, Engineering and Construction, 16–17 May, Singapore, 1996, pp. 361–368.
[31] J. Underwood, M. Alshawi, G. Aouad, T. Child, I. Faraj, The definition of design element specifications dynamically via a product supplier database Web-site, W78 Conference, Canada, June, 1999.
[32] N.-J. Yau, J.W. Melin, J.H. Garrett, S. Kim, An environment for integrating building design, construction scheduling and cost estimating, ASCE Seventh Conference on Computing in Civil Engineering and Symposium on Databases, Washington, DC, May 6–8, 1991.
[33] A. Zarli, V. Amar, Integrating STEP and CORBA for applications interoperability in the future virtual enterprises computer-based infrastructures, Proceedings of Intelligent Information Systems, Bahama, December 1997, pp. 309–315.