The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

VERSIONING OF IFC-BASED INFORMATION MODELS FOR COLLABORATIVE DESIGN

Aras Jalal Jaly Zada

Thesis submitted to the University of Nottingham for the degree of Doctor of Philosophy (PhD)

February 2016

Abstract

The engineering design process is an iterative and distributed process. It is often characterized by multi-disciplinary teams in multiple places working together, on a single project, using different models and software tools. The current collaboration approaches in AEC industry often focus on integrating and managing multiple models from multi-designers. Building Information Modelling (BIM) is playing a major role in facilitating collaboration. BIM provides an opportunity to electronically model and manage the vast amount of information embedded in a building project, from conception to completion.

In the design process, changes in the models are inevitable and very common. They can occur at any stage of the project, from different sources, and for various reasons, which can have significant effects on the process. Versioning is a solution for design change management. Many model versions can be created and distributed among the disciplinary teams. Despite many researches done on the subject and the availability of some software applications that deal with changes in design, the mechanism to cope with the changes among different model versions needs further studies to increase the management efficiencies and ensure designers have an up-to-date version of the model.

The challenges of information exchange in project management result from: a combination of the enormous amounts of information generated, the large variety of design systems involved, and data format utilised by different disciplines at various design stages. The Industry Foundation Classes (IFC) standard represents a paradigm shift for data and information exchange. The main goal for developing the IFC model is to provide a neutral data format to exchange information among different software programs. IFC models reflect the current state of BIM model. They do not take into consideration the process and results of latest changes among different BIM users nor record the history of earlier changes. This research work therefore investigates how to improve the process of managing the design changes from different disciplinary models.

This research developed a collaborative methodology to manage the design changes in different models. It tackles the challenges of the versioning process as a change management approach. This is done through extending the capability of the existing IFC schema to control and manage different design changes in different BIM models. The proposed extended IFC incorporates the changed information of the latest model version and provides the complete history of changes of all earlier model versions. A prototype system was developed in this research to implement and validate the extended IFC and to demonstrate using it to improve the management of the whole design process.

The research process involved undertaking a literature review to identify knowledge gaps and challenges in the areas of the (design process, BIM, IFC, and change management). The research also investigated and analysed the IFC standard and identified two key requirements of extending the IFC and implementing the prototype. Further aspects of this research include developing a framework to facilitate a collaborative design, extending the existing IFC schema, designing and implementing the prototype based on the extension, and validating and evaluating the extended IFC and the prototype system.

The research concludes that the extended IFC to handle versioning can effectively improve collaborative design. It addressed concepts involving comparing, storing, classifying, extending managing, versioning, exchanging, and sharing of modelling information in a collaborative way. The proposed process of managing design changes covered an important gap associated with current IFC models, which can be incorporated in future releases of the IFC standard.

Keywords: BIM, IFC, schema extension, versioning, collaboration, change management.

Acknowledgement

Firstly, I sincerely thank the Almighty God for seeing me through this stage of my life and I pray that He continues to direct and guide my footsteps.

This thesis would have not been possible without the help and support of many people and I would like to thank them profoundly. Firstly, I would like to express my deep gratitude and thanks to my supervisor Dr Walid Tizani for his kind support, guidance, and encouragement throughout the research period. In addition, I am grateful to Dr Akponanabofa Henry Oti who helped to spur the early stages of my work during information gathering and background research. I very much appreciate the academics and students of the Centre for Structural Engineering and Informatics (CSEI) research group for their interactions and constructive criticisms. I express my sincere gratefulness to my sponsors, Human Capacity Development Program (HCDP)/ Kurdistan Regional Government/ Kurdistan/ Iraq.

I will forever remain in debt to my mum for her support, sacrifice, and blessing over the years. Dad, although you are no longer with me, I will never forget that you always believed in me. Finally, heartfelt thanks go to my wife, Chnar, and my children, Shad, Shara, and Shna who without their support, encouragement, understanding, and patience, I would have never been able to present this thesis. I will never forget the lonely nights, health problems, and hard times they endured during the last couple of years.

Table of Contents

List of Figures

List of Tables

List of abbreviation

2D	Two Dimensional
3D	Three Dimensional
AC	Available Changes
AEC	Architectural, Engineering and Construction
API	application-programming interface
BIM	Building Information Modelling
CAD	Computer Aided Design
CIS	Construction Integration Standards
CSCD	Computer Supported Collaborative Design
CVP	collaborative versioning prototype
DBMS	database management system
DMS	Document Management Systems
E_V	Element Versioning
F_V	Feature Versioning
G_V	Global Version
GUID	Global Unique Identifier
Hv	History Versioning
IaaS	Infrastructure as a Service
IFC	Industry Foundation Classes
ISO	International Organisation for Standards
L_V	Local Version
OOP	Object-Oriented Programming
PaaS	Platform as a Service
PDMS	product data management system
MC	Missing Changes
M_V	Model Versioning
NBIMS	National Building Information Model Standard
NIBS	National Institute of Building Science
SaaS	Software as a Service
SCM	software configuration management
SDK	Software Development Kit
SMC	Solibri Model Checker
STEP	Standard for Exchange of Product Data
V	Version
VCS	Version control system
XML	Extensible Markup Language

Chapter 1

Introduction

1.1 Background

Building design is a distributed and continuous iterative process. Multi-disciplinary teams and experts are involved in this process and there is a large amount of information transfer between them and at each of the design stages. Collaboration among multi-disciplinary designers has a crucial impact on the success of any engineering project in terms budgets and time.

Each disciplinary designer works on a different model associated with different aspects of the building depending on his/her speciality. These models are often large, complex and shared in certain parts. Unavoidably, these models keep changing and increasing in size and complexity as more information is added and modified. Design changes are very common in AEC projects and usually happen at any stage of the project, from different sources, for various reasons, and can have significant effects. Successful management of the design changes has a great influence on the success of any engineering project (Motawa et al., 2007).

Collaboration between disciplinary teams has revolved around the exchange of 2D CAD drawings and 3D CAD models for many years, which does not promote a true collaborative approach (Wu and Hsieh, 2012). With the emergence of Building Information Modelling (BIM), the AEC industry has been changed dramatically. BIM is envisaged to play an important role in the process of collaborative design by providing an opportunity for the exchange of intelligent models. Although most of designers models can be prepared using conventional CAD software, BIM software produces these models more efficiently, and maintains consistency at all times for the same model (Azhar et al., 2011). Financial savings because of BIM have been documented worldwide. According to the SmartMarket report on the value of BIM (McGRAW-HILL, 2014), 74% of Western European BIM users and 71% of North American BIM users reported returns on their investment in BIM.

It is difficult to expect that a single BIM application would meet all the requirements of the AEC industry. For the purposes of collaboration, it is essential to seamlessly transfer information between different AEC applications effectively, allowing applications to utilize the information created by other applications (Wang et al., 2009). Hence, interoperability of design software and standardization of building data is a major requirement for exchanging information across multiple software applications and for successful collaboration (Tizani and Mawdesley, 2011). The most prevalent data exchange standard is the Industry Foundation Classes (IFC). It was specifically developed as a means to exchange model-based data among model-based applications (Steel et al., 2012b). The IFC schema defines a standardised data structure that can be used as a mechanism for sharing semantically rich building information among applications. It is formally adopted worldwide as it can describe multi-disciplinary information.

The fact that many disciplinary teams are involved in the design process, using different BIM models, managing changes in these models, and maintaining consistency among them are a challenge to the success of any engineering project in the AEC industry. The next section starts with the discussion of the limitations and the challenges of the contemporary design process.

1.2 Problems Statement in the Design Process

The design process in an engineering project is a complicated activity involving multi-disciplinary designers that are using different models and applications. Although this process has been presented in a linear manner, in practice, designers frequently jump back and forth between design stages to redesign some information of the model or to improve what they created before, which often leads to complication of the process (Tunstall, 2006). There are some difficulties in managing the design process due to the different competency and responsibility of designers and the complexity of the models. Collaborative design requires a wider range of contributions to the design process. It often requires additional and significant commitment among participants that go beyond conventional technical roles. It needs sophisticated schemes to solve clashes, manage changes and achieve better system performance (Taylor, 2009). The efficiency of the current building design process is prevented from reaching its real potential. This is largely due to the limitations in the use of non-integrated design systems and the lack of effective collaborative platform (Ruikar et al., 2005).

It is recognized that the adoption of BIM could result in many benefits to the design process. Notwithstanding, the move from traditional CAD systems towards adopting BIM comes with many challenges in information management as BIM is not yet fully matured (NIBS, 2007). The huge information that is embedded in each disciplinary BIM model and the wide number of BIM applications that are used by each disciplinary team do not allow for the smooth transfer and management of the BIM models between processes and disciplines. This makes design changes inevitable due to the creative and iterative nature of design and specialist variations of the participants in the design process (Gareis, 2010). Design changes by any designer are usually made in isolation from other-disciplinary teams and even from the same-disciplinary team. Changes in BIM information by any designer must be clarified to others to ensure consistency (Macdonald, 2013). BIM models contain shared information with other disciplinary teams (e.g. beams, columns, etc.) and/or specific information with the same team (e.g. reinforcing steel bars, cooling ducts, etc.). Therefore, changes made to the model by any discipline may affect all/some/or none of the other disciplinary designers (Jaly-Zada et al., 2014). The shared information among different disciplinary designers requires good management in order to positively influence the collaborative design process.

With the purpose of dealing with the evolving information, elucidating of track changes, supporting of retracing change history, and ensuring consistency of distribution models, versioning technique is a solution of the design change management (Koch and Firmenich, 2011, Kim et al., 2011). Therefore, many model versions can be created and distributed among the disciplinary teams. The current process of comparing two large-scale model versions by each designer to find the design changes is time-consuming and cumbersome. It is even more difficult when the models for multi-disciplinary designers are being updated frequently to meet the project requirements (Turner, 2014). Even with the availability of some software applications to deal with the design changes, the process is still neither efficient nor comprehensive.

For clarification, the expression “element” has been used in the work to describe the objects that represent the building (e.g. column, beam, wall, door, etc.…). Moreover, the expression “feature” has been used to define the properties (e.g. geometry, locations, etc.…) that form an element and the characteristics (e.g. cost, analytical results, etc.…) that belong to an element. The general trend with the design process is versioning the design information at the model level. Essentially, designers do not want to know that a new model version has been issued. While, the intention is to know what building elements (e.g. beams column, etc.…) or its features (e.g. geometry, locations, cost, etc.…) of the new model version have been changed in a way that the design changes and the design history “old evolution” of the information in the new model version can be easily extracted and used. Below are some limitations in the conventional design process and in the current software and applications used to manage the design changes:

- The changed information in the model by one designer is not clarified and recognized for the others who receive the new model. Each disciplinary designer needs to compare the new model version with the old model version to identify these changes. This process is a replicated by all participants.

- The comparison process might also identify changes that are superfluous or not required by the designer who receive the new model version.
- The designers need to use specific and private applications to display the new changes numerically and graphically.
- Each designer has to deal with a set of sequential model version files. Therefore, the new with all the previous model versions have to be saved in a secure and easily accessible location.
- To find the evolution that has happened about specific information (building element, e.g. beam, column, etc.…, or specific feature e.g. geometry, locations, etc.…" of that element), the designer has to go through all the model version files starting from the first to the current version.
- All the disciplinary teams use a shared central storage to store many model version files. Moreover, shared and specific information of the model for each of the disciplinary teams are stored in the shared files and thus lead to much redundant information for the disciplines.
- Versioning the BIM information at the model level to deal with the whole model information without taking into consideration versioning the elements or their features make the process of finding and identifying the change information difficult.

One of the main characteristic of BIM is its ability to exchange information across multiple software applications (Muller et al., 2015). IFC is the most wide-ranging and wide-spreading data model to exchange information in the AEC industry (Rezgui et al., 2013, Zhang et al., 2015b). However, the IFC standard cannot yet describe all the concepts necessary in the AEC industry. Researchers have been looking at expanding IFC standard scopes in different domains, such as sustainability, structural engineering, cost, etc. Traditionally, the IFC standard only reflects the current state of information (Gökçe et al., 2013). The IFC model developers' main goal is to provide a neutral data format to exchange data among different software programs. It is not developed to deal with changes to BIM models. The IFC does not specify a means for sharing the affected changes among different BIM models nor the recording of the history of earlier changes. The versioning technique is not covered within the concept of the IFC standard. Thus, the existing gap of not being able to manage design changes remains a challenge.

1.3 Research motivation

Even with the sophistication of characterizing BIM environment, BIM is still not at a fully matured stage. The fact that many of the disciplines involved in the design process, using different BIM models, releasing multiple model versions, dealing with different design changes and trying to maintain consistency among them are still a challenge in the AEC industry. Collaboration among multi-disciplinary designers during all design stages is the key to achieving better design solution and project success. The improvement to collaborative working in the AEC industry has been highlighted in a number of standards, reports and publications, such as (BS-1192, 2007), (Zimmerman and Eng, 2009), (Filippi and Cristofolini, 2009), (Cabinet-office, 2011), (Thomassen, 2011), (Kassem et al., 2014), and (Tomasowa, 2015). Therefore, the awareness of the needs of collaborative design practices within a building design environment is widely recognised.

The advanced development in the ICT sector and the high abilities in BIM performance have not prevented from making changes in design models by any discipline. Lack or delay dealing or managing the changes have a significant impact on the cost, time, quality, and performance of a project.

Further, design changes are a common issue in each engineering project, at any design stage, encompasses all disciplinary designers and models. Therefore, managing the change information is a worldwide challenge that requires high level of collaboration.

AEC industry is one of the largest economic sectors of any industrialised country. The performance of this industry is very important to the governments as well as to those within the industry. Hence, the AEC industry is certainly concerned about devising means and methods to improve this industry. In the UK, the government has mandated that all centrally funded work is to be undertaken using BIM by 2016 across their projects in order to meet the industry challenge of reaching full collaboration among all participants (Cabinet-office, 2011, Rezgui et al., 2013). Therefore, collaborative BIM is becoming a prerequisite for all projects in the UK by 2016.

1.4 Research Questions

The observed limitations of the current design process in dealing with different changes in different BIM models and the requirements to build a robust collaborative design process have led to the formulation of main and sub-research questions in this thesis.

- **Main-question: How can a BIM strategy be employed to manage design changes to better support multi-disciplinary collaborative design?**

A good way to achieve this is to version all design changes in a manageable manner and clarify them to all design disciplines through using and extending one of the open data exchange standards. The broad research question above can be further decomposed into two sub-questions.

- **Sub-question: How can the extension to an existing data exchange standard support the management of design changes?**

The benefits of the extended model above can be used to improve the whole design process through implementing the extension to deal with multi-disciplinary teams, different BIM models, and central storage server. This leads to another sub-question of this thesis:

- **Sub-question: How can the proposed extension be verified and validated?**

This involved the utilization of programming and modelling techniques that takes advantage of contemporary object-oriented language such as C# to capture and define multi-disciplinary versioning of information and creating requisite interface with a BIM enable environment.

1.5 Aim and objectives

The aim of this research is to develop a methodology to improve the collaborative design process by versioning the design changes of different building information models based on a data exchange standard. The objectives of this thesis are as follows (Figure 1.1):

1. Investigate the state of the art and identify shortcomings in managing design changes in building information models (BIMs) and data exchange standard (IFC).
2. Identify the requirements for modelling a collaborative design process using the versioning concept in the IFC standard.
3. Propose a collaboration versioning framework to capture various factors influencing the versioning concept to enhance the collaborative design process.
4. Implement and demonstrate the versioning concept in the IFC standard through extending the IFC standard.

5. Implement and demonstrate a collaboration versioning prototype based on the proposed framework and the IFC extension.
6. Feedback on the effectiveness and efficiency of the proposed IFC extension and the prototype software.

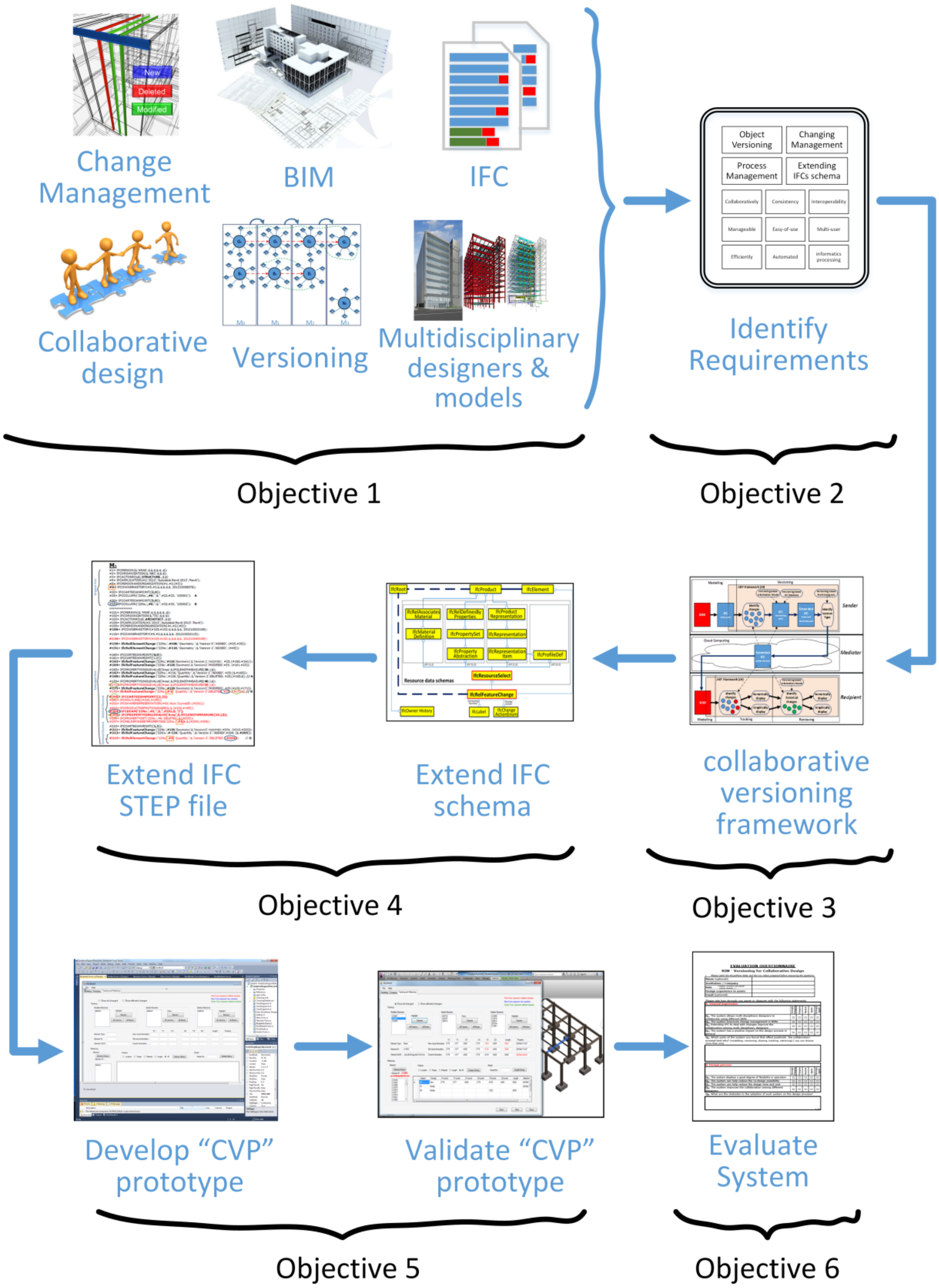


Figure 1.1 Research objectives

1.6 Methodology

The techniques that have been used for collecting, analyzing and interpreting information are a combination of different research methods adopted to fulfill the objectives within a research project, such as quantitative, qualitative, process flow, development, programming, software engineering, and feedback methods for extending the IFC standard and developing a prototype to support collaborative design requirements.

- **Objective 1:** Investigate the state of the art and identify shortcomings in managing design changes in building information models (BIMs) and data exchange standard (IFC).

The objective focused on understanding the current collaborative design process as a whole from the point of view of the building design disciplines and information technology industry and assessing the current approach for managing the design changes, precisely on BIM and IFC.

To meet the goals of this objective, the author carried out a comprehensive review of relevant literature taken from several resources including textbooks, journals and conference papers, international organization reports, research thesis, webinars and lectures to meet this objective which constitutes the broad research area that helped in identifying research gaps.

- **Objective 2:** Identify the requirements for modelling a collaborative design process using the versioning concept in the IFC standard.

After gathering the essential information and identifying the problems from the previous objective, the main requirements and specifications for managing the design changes have drawn in two aspects. The first aspect entails requirements for extending the scope of the IFC standard to deal with the design changes and the versioning concept. The second aspect includes requirements for developing a prototype to implement the extended IFC.

- **Objective 3:** Propose a collaboration versioning framework to capture various factors influencing the versioning concept to enhance the collaborative design process.

This objective is targeted at developing a framework of abstractions made from the requirements of the previous objective. Integrating the versioning process within IFC standard to describe the design changes in a neutral data model has been elaborated to design a collaboration versioning framework to provide a foundation for managing changes in different BIMs and collaborating among multi-disciplinary designers.

- **Objective 4:** Implement and demonstrate the versioning concept in the IFC standard through extending the IFC standard.

The requirements stage (objective 2) and the design stage (objective 3) served in building the next stage, which is the implementation stage. This objective described the first implementation part that is related to extending the IFC standard to include the concept of versioning. To meet the goal of this objective, factors related to the versioning concept were added to the IFC-EXPRESS schema and then executed these extensions in the IFC-STEP file. A case study has presented to validate the implementation of the versioning concept in the IFC standard.

- **Objective 5:** Implement and demonstrate a collaboration versioning prototype based on the proposed framework and the IFC extension.

The second implementation part entailed developing a collaboration versioning prototype (CVP) to verify the feasibility and usefulness of the versioning concept in an extended IFC model. C# programming language of Microsoft .NET was found suitable for this implementation as it interacts with the BIM applications easily using the application-programming interface (API). A typical case study was presented to validate the use of the proposed prototype system in a typical design activity.

- **Objective 6:** Feedback on the effectiveness and efficiency of the proposed IFC extension and the prototype software.

The extended IFC and the prototype software were feedbacked using a number of methods to study its efficiency to support the collaborative design requirements. A set of questionnaire were prepared and demonstrated to a group of carefully selected feedbackers that have been employed to carry out the feedback stage of the prototype software.

1.7 Structure of the Thesis

The thesis is composed of eight chapters and appendices. Below is a brief description of these chapters:

- **Chapter 1: Introduction.**

This chapter presents the research background and motivation, the research questions, aim and objectives, and the research methodology.

- **Chapter 2: Design Collaboration, BIM and Change Management.**

This chapter is divided into four main sections. It starts with reviewing design stages and clarifying how collaboration is essential for the success of engineering projects. It continues with the review of BIM adoption in the design process. It then provides an overview of managing design changes and versioning concepts. Some of the main limitations of integrating BIM in the design process are also discussed.

- **Chapter 3: Industry Foundation Classes for Interoperability.**

This chapter reviews the IFC environment, in general. It starts with reviewing the interoperability in BIM and different data exchange standardisation. It provides an overview of the IFC evolution and then demonstrates the EXPRESS data definition language and the STEP physical file format. Finally, the limitations of managing design changes are discussed. Chapter 2 and 3 represent *objective 1* in this research.

- **Chapter 4: Collaboration Versioning Methodology.**

This chapter presents the key requirement for adopting the versioning concept through extending the scope of the IFC and developing a prototype to implement the extended IFC *(objective 2)*. Moreover, it then designs a collaboration versioning framework to provide a solution through using the versioning concept in the extended IFC with the information model and among the disciplinary designers *(objective 3)*.

- **Chapter 5: IFC–based Implementation of the Versioning Concept**

In this chapter, extending the IFC standard is covered to provide details on integrating the process involved in the versioning concept into the IFC standard. A case study is presented to explain further the implementation of the versioning concept in the IFC-STEP file. This chapter represents *objective 4* in this research.

- **Chapter 6: Prototype and Demonstration**

Discussions on the implementation of the prototype are presented in this chapter to verify the usefulness of extending IFC to deal with the versioning concept. It also outlines the testing and validation of the proposed prototype. This chapter represents *objective 5* in this research.

- **Chapter 7: Industrial Feedback**

This chapter presents the feedback results of the proposed extension to the IFC and of the prototype application. It presents the aim of getting feedback process and the methodology. It also discusses aspects related to the development of the questionnaire and finally discusses the results of the feedback. This chapter represents *objective 6* in this research.

- **Chapter 8: Conclusion and Recommendations**

This chapter highlights the main research findings and the possible future research tracks. The chapter also states the conclusions and the main research contributions to knowledge.

1.8 Summary

This chapter laid the foundations for the research work. It presented the research background, formulation of the research problem, research motivation, research questions, aim and objectives, methodologies, and the structure of the thesis.

Chapter 2

Design Collaboration, BIM and Change Management

2.1 Introduction

Building design is a process that comprises a series of discrete activities in a period of time, with the participation of experts from varying professional domains. Building information modelling (BIM) is one of the most promising recent developments in the AEC industry. Employing BIM enhanced the building design process by introducing new avenues for collaboration. Design changes of the building models are unavoidable even with the contemporary BIM technology, which need to be managed and coordinated as explored in this chapter.

The chapter is divided into four main sections. It starts with reviewing the current design process and clarifying how collaboration is essential for the success of engineering projects (Section 2.2). It continues with the review of BIM adoption in the design process (Section 2.3). The chapter then provides an overview of possible changes effected on the building models during the design process and how these changes can be managed (Section 2.4). Some of the main challenges of integrating BIM in the design process are then discussed in (Section 2.5).

2.2 Design Collaboration

Collaborative engineering is a concept used to describe the process of improving the engineering design with the objectives of better product quality, reduction in cost, shorter lead-time, and higher customer satisfaction (Shen et al., 2008). Generally, a design process is a knowledge intensive and collaborative activity. Collaboration during the design stage is essential to ensure the success of any engineering project (Girard and Robin, 2006).

The next sub-sections provide an overview of the design stages, discuss collaboration during the design process, and it is followed by examining the role of information technology in the collaborative design process.

2.2.1 Design Stages

Building design is a comprehensive and continuous iterative process. It includes a wide range of issues that require technical details and engineering expertise. The traditional building design process, which is also known as sequential design, is often divided into a series of phases or stages. Each stage of the development process, where different ideas are successively refined, focuses on specific aspects of the design. The next design stage cannot start until the completion of the previous stage. This means that the information flow is highly one directional (Clarkson and Eckert, 2004).

There are sequential stages to the design process that may vary for particular projects or design disciplines. In the early stages, emphasis is on making high-level decisions with major impacts that truncates the options available during later stages. The later phases are spent on exploring and developing selected options and filling in needed details. Below is a brief discussion of the typical design stages according to earlier works (Autodesk, 2012, DYERGRIMES, 2010, Fahdah, 2008, Pahl et al., 2007):

- **Pre-design stage**: The processing of activities to design any building project starts with the pre-design stage to (i) sort out what the project might be, (ii) study the constraints and opportunities of the proposed site, (iii) roughly schedule the allocated budget, (iv) plan and clarify the tasks and (v) elaborate the requirements.
- **Conceptual design stage**: The architect usually starts the conceptual design stage (also called schematic design stage) by working with different ideas for the building within the framework of the client requirements. The architect typically evaluates several rough concepts for the overall shape, massing, orientation, and positioning of the building on the project site. At this stage, the abstract concepts of the project outlines and the distribution of project tasks among multi-disciplinary design teams will be studied and evaluated. It is often carried out with little detail about the precise features of the building envelope or the configuration of the rooms inside.
- **Preliminary design stage**: the architectural team chooses the concept design that best suits the requirements. The team then develops the concept into the preliminary layout. 2D plans and elevations with the possibility of providing 3D drawings are prepared using different building design software (for instance, AutoCAD and Revit). The

architect then passes on the preliminary design to the other disciplinary designers. Preliminary structural and services drawings of the building are then prepared based on the architectural design. For example, an initial set of drawings is prepared by the structural team that includes information about forms, locations and materials. Drawings about individual structural members are prepared in the next stage.

This stage is an increasingly collaborative process that involves teams of disciplinary designers coming together, and discussing options, opportunities, constraints, and requirements to minimize any future conflicts. At the end of this stage, the Initial cost for the project is calculated based on the preliminary drawings and compared with the project budget to pave the way for the next stage. Fundamental changes to the design concept after this point will be significant in terms of costs and will inevitably affect subsequent program timings (Tunstall, 2006).

- **Developed design stage**: After approving the preliminary drawings, the different disciplines will advance and develop the initial design into a more complete design. At this stage, the ideas and design features that have been selected during the preliminary design phase are developed and explored more collaboratively. For example, the structural team analyzes and designs the building model and needs to prepare a set of drawings including plans and sections for the foundations, columns, beams, slabs, walls, stairs, etc.….

Typically, all designers work in parallel with regular exchange of drawings and information. These drawings are usually prepared in the form of 2D plans and 3D models. At the end of this stage, all disciplines will need to finalize their drawings for the estimated cost to be updated and refined.

- **Detailed design stage**: Following the agreement on the developed design, all disciplinary teams will produce full construction specifications, drawings and schedules, which guide contractors in construction detailing, fabrication, and installation. It includes site works and finishes for the architect and the description and location of all reinforcement in the drawings (reinforcement drawings) with full schedules for the disciplinary teams. The detailed drawings are usually called “shop drawings”. Technical and quality specifications also form part of the detailed design stage to ensure that the project requirements are clearly formulated for the contractor. Design work becomes more specialized at this stage with a minimum exchanging of information among the different disciplines. By the end of this stage, the completed drawings are submitted to the quantity surveyors for material estimation and final costing.

RIBA Plan of Work 2013 has been a bedrock document for organizing the design, construction, maintenance and operation processes (Sinclair, 2013). It was published by the Royal Institute of British Architects (RIBA) to provide a shared framework that offers both a process map and a management tool. It is the most widely used framework for the delivery of construction projects within the UK and elsewhere that divided the building projects into eight work stages (0-7) (Dawood et al., 2013). The main benefits of RIBA Plan of Work 2013 can be summarized as:

- Specifics the tasks and outputs required at each stage that may vary or overlap to uniform specific project requirements.
- Performances as guidance for the preparation of detailed professional services and building contracts.
- Acts across the full range of sectors and project sizes.
- Provides straightforward mapping for all forms of procurement.
- Integrates sustainable design processes.

- Maps Building Information Modelling (BIM) processes.
- Provides flexibility in relation to planning procedures

McElroy and W.Tizani (2008) presented a process map to facilitate the integration of architects and structural engineers, and to show the need for a documented approach which helps all the designers to understand their roles throughout the design process. The design process has been divided in this paper into five stages (understanding the project, pre-planning design, scheme design, detailed design and production information). Ruikar et al. (2001) proposed a re-engineered design process to improve the quality of the design by supporting integrated design. This study developed a process model that supports an underlying product model in the re-engineered building design process. Adamson (2006) focused on the development of the designer's relationship standardization throughout the design work, concentrating on customer requirement. The author illustrated the need to define the integrated project process to improve the quality and efficiency of the UK construction industry. Ruikar et al. (2005) stated that the design process have three main aims; technical, architectural and financial aims. They developed an As-Is process model to highlight the inefficiencies within the current design process and to demonstrate typical information flow between the designers involved in the structural design process.

2.2.2 Collaborative Design

There is no precise definitions for the term “Collaboration” in the AEC industry, although it has been the subject of the research in academia and practice for many years (Xiaolong et al., 2007). Xue et al. (2010) defined it as a joint working among various stakeholders, multi-disciplinary teams and different organizations to effectively and efficiently accomplish a project. Collaboration can be thought of as “joint problem solving”

(Roschelle and Teasley, 1995). There is an enormous range of terms describing collaboration in the engineering industry, such as cooperation, alliance, partnership, coalition, teamwork, joint venture, and coordination (Grilo and Jardim-Goncalves, 2010).

Littler et al. (1995) distinguished many factors that affect the successful collaborative process. These factors include people, vision, trust, time, planning, communication, learning together, decision-making, technology, and flexibility. Bouchlaghem (2012) explained that in order to achieve an effective collaboration in the building design, it is required from the designers to succeed in issues such as, coordination, creativity, engagement, good communication, efficient exchange of knowledge, shared vision and objectives, management activities and responsibilities, and adopting joint methods and procedures. Collaborative design recruits a wider range of contributions to the design process (Xiaolong et al., 2012). It often requires more and high commitment among participants that go beyond conventional technical roles (Bouchlaghem, 2012). Tizani et al. (2002) pointed out that an important step towards achieving collaborative design is in finding a way for all involved parties in the AEC project to be able to express their intentions and decisions, regarding the design and construction of the building, in a compatible manner.

There are different types of collaboration among the members in a project. Where the collaborative design team often work independently and in parallel using different engineering tools distributed in separate locations, even across different time zones around the world, the design process may be called distributed collaborative design (Chen et al., 2005b). Anumba et al. (2002) identified four different types of collaboration related to the co-located and distributed environments:

1. **Face-to-Face Collaboration** (same time–same place): it usually involves working in the same locality such as sharing the same design office where participants interact in real time discussions. Traditionally, design processes are still favoring this type of work environment.

2. **Asynchronous Collaboration** (different time–same place): it carried out in a shared place while the participants do not communicate at the same time they do so via electronic mediums and a notice / bulletin boards.

3. **Synchronous Distributed Collaboration** (same time–different place): it involves real-time collaboration between participants over the Internet which are located far apart to jointly work, retrieve, filter, search, and organize the design. Participants are associated via any of a vast array of communication technologies such as telephone, computer-mediated conferencing, video conferencing, electronic discussion medium (e.g. Skype).

4. **Asynchronous Distributed Collaboration** (different time–different place): it conducted from dispersed locations and not in real - time communication between participants (collaboration separated in time). Different communication media are used, which allows for passing of ample time between sending and receiving of information, such as fax machines, electronic mail, and cloud storage services (e.g. Dropbox), etc.

Johansen (1988) used 2D time-space matrix to study cooperative works. The matrix classifies collaboration into synchronous and asynchronous patterns; this matrix cannot fully represent the emerging trends of collaboration. Chen et al. (2005b) extended the matrix to a 3D time-location-group to describe when, where and who are collaborating. According to the functions and roles of the users involved in a collaborative design activity, collaboration could be structured as either a horizontal or a vertical way. The horizontal collaboration puts the focus on sharing a design team from the same or different disciplines to perform a regular task. The vertical collaboration can establish an effective channel of communication between the upstream design and the downstream manufacturing tools. Due to these different levels of collaboration and interaction between users, Li et al. (2005) categorized collaboration into three types, visualization-based collaboration, co-design collaboration, and concurrent engineering-based collaboration.

2.2.3 The Role of Information Technology in Collaborative Design

Collaborative design has progressed significantly based on the rapid advancement of information and communication technologies. Application of collaborative design that is called Computer Supported Collaborative Design (CSCD) has become more promising (Shen et al., 2008). Saad and Maher (1996) suggested that computer-support for collaborative design requires a common understanding of the design artifact between the design teams. Li and Qiu (2007) clarified, that computer-aided design (CAD) and product data management (PDM) are seen as two primary pillars to support collaborative design. CAD is a tool that focuses on the design to provide a platform for embodying design models and drawings, and PDM is a tool that focuses on the process to organize the information, communication, and collaboration between designers. Smith and Tizani (2006) proposed an integrated system through implementing information and process model in a prototype software that is used by multi-disciplinary teams in collaborative building design process.

In recent times, some researchers began to investigate internet supported collaborative design technologies. Zimring et al. (2001) presented a collaborative on-line studio (CoOL Studio) to provide access to on-line cases and to support the teamwork among the participants. Shyamsundar and Gadh (2002) developed an integrated interfaces for viewing product assembly over the Internet to make collaboration faster and easier during product assembly design. Roshani and Tizani (2005) proposed a collaborative AEC design environment “CODE” system to integrate the work of distributed project participants. This study has described collaboration as a function of time, space and shared working environment, which allowed the use of real-time design tools over the World Wide Web (WWW). Amarnath et al. (2011) emphasized, that using design applications in the cloud can further enhance project integration through achieving the optimal levels of collaboration, coordination and communication.

There is increasing interest in synchronous and asynchronous collaboration among multidisciplinary designers. Fahdah and Tizani (2008) employed distributed information technologies to enhance the processes of synchronous collaborative building design through proposing a product model that the distributed designers can work concurrently on a centralized shared model. Gu et al. (2011) analyzed the effect of (a) 3D virtual, for supporting remote design collaboration, and (b) tangible user interfaces, for enhancing co-located design collaboration, on synchronous design collaboration while the designers are remotely located. Flurl et al. (2012) developed a collaboration platform for the interactive planning to enable different planners to work synchronously and asynchronously and thereby using multi-scale planning data. The collaboration platform provided the possibilities to visualize the planning process on portable devices.

2.3 Building information modelling (BIM)

Building Information Modelling (BIM) is the process of generating and managing data and information about a building during its entire life cycle from concept design to decommissioning (Howard and Björk, 2008). BIM has gained widespread attention in the AEC industry (Dawood and Iqbal, 2010). The majority of disciplinary designers and construction firms have their transitions from CAD to BIM well underway (Kensek, 2014).

The next sub-section starts with discussing the evolution from CAD to BIM approaches. It then reviews the role of BIM in the design process, followed by the integration of BIM with cloud computing. Finally, the UK strategy to adopting BIM in the AEC industry is discussed.

2.3.1 From CAD to BIM

The first idea of computer modelling of buildings was proposed in 1960s (Ricci, 1973, Bijl and Shawcross, 1975). In the mid-1970s, technology vendors were gradually replacing the traditional “paper & pencil” process with workstations running computer-aided design (CAD) applications (Robertson and Allen, 1993). Design teams started to provide more capabilities to reproduce manual drafting with electronic drafting. CAD drawings, which are usually created from 2D components (such as lines, hatches and text, etc.), and 3D geometry models (such as contour, surface, and solid models), imitate the traditional paper drawings. Each component in both drawing approaches is created independently, so design changes need to be followed up and implemented manually (Azhar et al., 2008). The various operations in developing drawings under these approaches are at a relatively low level. According to this manual process, it shows a weakness in terms of using it in massive and complicated projects as well as the possibility of errors (Weygant, 2011).

Solid modelling became the most advanced method of geometric modelling in 3D to represent objects. Building modelling based on 3D solid modelling was first developed in the late 1970s and early 1980s (Cerovsek, 2011). There are three popular ways to represent 3D solid models, decomposition models, construction models, and boundary models. The last two models are more common and are described below:

- Construction models: often known as Constructive solid geometry (CSG). It uses a tree of Boolean operators (union, intersection, and difference) to define the shape of the desired 3D object.
- Boundary models: often abbreviated as boundary representation approach "B-rep" or "BREP". It is represented as a volume contained in a set of faces together with topological information, which defines the relationships between the faces.

When object-based parametric modelling was first released in the 1980s, it did revolutionise the CAD industry. It fundamentally changed the way that engineering organisations developed 3D models, and changed the designs as well. It does not represent objects with fixed geometry and properties. Rather, it represents objects by parameters and rules. By defining different parameters and rules, objects can be controlled on how they act and interact depending on the context (Weygant, 2011). In object-based parametric modelling, instead of designing a building element "like a beam", model families are defined. These parametric object families are a set of rules and relationships. Parameters (such as length, height, and angles) are constricted by the family rules. Example of rules can be that the object has to be *attached to, parallel to* and *distanced from* another object. Most allow "if-then" conditions. This means that each object in family differs based on their parameters and context. The current generation of building modelling tools is the outgrowth from four decades of research and development on computer tools for interactive 3D design, culminating in object-based parametric modelling (Eastman et al., 2011b).

2.3.2 BIM

Building information modelling (BIM) has introduced a certain amount of parametric modelling into mainstream building design. Current BIM tools are essentially parametric models of the building artefact composed of predefined object families with properties, behaviour, and rules (Boeykens, 2012). It provides a large number of default parametric dependencies, such as relationships between the coordinate axes of a building and the position of components or interactions between wall heights and roof shapes (Ignatova et al., 2015). The modelling procedure with BIM goes from being a geometric design and becomes a multi-layered, data rich and intelligent representation of a structure. For instance, every change made to an element in BIM is automatically propagated through the model to keep all components, views, and annotations consistent. This eases collaboration between teams and ensures that all information (floor areas, schedules, etc.) are updated dynamically when changes in the model are made. BIM users can define much more complex structures of object families and relations among them compared with CAD systems and without undertaking programming-level software development. For instance, a wall attached to columns and slabs can be defined with BIM from scratch by a knowledgeable nonprogrammer (Eastman et al., 2011b). Weygant (2011) simplify the relation between the CAD and BIM technologies into, CAD + specifications = BIM.

BIM applications imitate the real building process. Instead of creating drawings from 2D line-work, buildings are virtually modelled from smart construction elements such as columns, walls, slabs, windows, etc... (Arayici, 2015). It allows for the creation of intelligent objects that can be used to construct a building as an assembly (Cinelis, 2015). Since most processes in BIM are automated and the participation of human resources is reduced, it is claimed that by using BIM, the efficiency of collaborating, monitoring, controlling, and updating in construction projects' life cycle is improved remarkably (Namini et al., 2011). For instance, in CAD models, a column change may be edited in the plan and elevation, but section detailing may have been overlooked, thus creating a conflict in the documents. In BIM models, changing in any view will automatically change the column information in all other views, as well as all plans, 3D model, and tables.

BIM is an ambiguous term that carries various definitions to different particular professionals (Amor et al., 2007). BIM is not only defined in several ways but some confusion exists at three different levels. Some professionals define BIM as a technology represented by software application, whereas for some it is a process for designing and documenting building information. Yet others define BIM to the level where it is a totally new approach to practice and promote the profession that requires the implementation of new policies, contracts, and relations amongst project stakeholders (Underwood, 2009). BIM is all of that. It is a mutually dependent network of policies, processes and technologies, which together form a procedure to manage the building design and construction in a digital way throughout the building's life-cycle (Succar, 2009, Kassem et al., 2014).

The National Building Information Model Standard (NBIMS), developed by the US based buildingSmart group, describes three scopes of BIM (NIBS, 2007): 1. a product or intelligent digital representation of data about a capital facility. 2. a collaborative process, which covers business drivers, automated process capabilities, and open information standards. 3. a facility lifecycle management tool of well-understood information exchanges, workflows, and procedures that teams use as repeatable, verifiable, transparent, and sustainable information based environment. BIM acts as an enabler of interoperability and is a facilitator of data sharing and exchange between software applications (Jaly-Zada and Tizani, 2014). BIM and interoperability issues are discussed in more details in the next chapter.

Table 2.1 Widely used terms relating to BIM(Succar, 2009).

Sample terms	**Organisation or Researcher Reference**
Asset Lifecycle Information System	Fully Integrated & Automated Technology
Building Information Modelling	Autodesk, Bentley Systems and others
Building Product Models	Charles Eastman
BuildingSMART™	International Alliance for Interoperability
Integrated Design Systems	International Council for Research and Innovation in Building and Construction (CIB)
Integrated Project Delivery	American Institute of Architects
nD Modelling	University of Salford — School of the Built Environment
Virtual Building™	Graphisoft
Virtual Design and Construction & 4D Product Models	Stanford University— Centre for Integrated Facility Engineering

Researchers had been investigating the mechanisms and implications of BIM for many years before the phrase “Building Information Modelling” was raised as a new term to describe virtual design, construction, and facilities management. The term "Building Product Models" was used in the US to describe the concept of BIM, while the term "Product Information Model" was used in Europe and when the European and US nomenclature amalgamated the term "Building Information Modelling” came to life in early 2002 (Eastman et al., 2011b). Succar (2009) collated the more widely used terms relating to Building Information Modelling in both research and industry works as shown in Table 2.1.

BIM is not an object or a type of software but a human activity that involves wide process changes in design and construction. For that, Eastman et al. (2011b) intentionally used the term BIM to describe an activity (building information modelling) instead of an object (building information model). Succar (2009) provided certain common connotations of multiple BIM terms as in Figure 2.1.

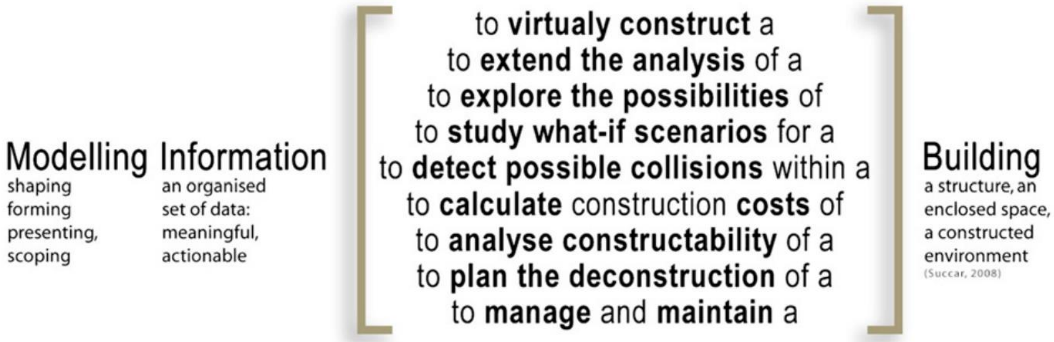


Figure 2.1 common connotations of the BIM term (Succar, 2009).

2.3.3 *The Role of BIM in the Design Process*

BIM is more than a technology or a tool. It represents a sea of change to the design process. BIM presents the opportunity of electronically modelling and managing the huge amount of information embedded in a building project, from its conception to end-of-life.

Efficient design management is important to enable designers to respond proficiently to the competitive construction industry and the clients' requirements. Employing BIM introduces a more integrated building design process due to the improved exchange of digital information compared to the traditional process. An important objective of the full lifecycle BIM initiative is to smoothen the flow of information and avoid data loss between the design phases (Al Hattab and Hamzeh, 2013). BIM is being increasingly integrated into collaborative design and modelling processes. It extends the project drawings beyond 3D models; it covers spatial relationships, geographic information, and quantities of building components (Kymmell, 2008).

Multi-disciplinary designers are working on different BIM models depending on the specialization of each one of them. Working within multi-BIM models needs nested and inherent collaboration among designers and between systems and applications to achieve the anticipated goal (Grilo and Jardim-Goncalves, 2010). Early collaboration has great benefits for the planning of a building project, and adoption of a BIM model is one of the best means to ensure the early and deep collaboration of the various and heterogeneous project team members (Jaly-Zada et al., 2015). A basic premise of BIM is collaboration by different stakeholders at different phases of the life cycle of a facility to insert, extract, update, or modify information in the BIM process to support and reflect the roles of that stakeholder (Hardin and McCool, 2015).

According to Eastman et al. (2011b), the role of BIM used in the design phase can be categorised in three viewpoints depending on their level of information development.

- Addresses the conceptual design, which involves generating the basic building plan, its massing and general appearance, determining the buildings placement and orientation, the structure, and how the project will realise the basic building program.
- Addresses the detailed design and analysis of the building system. Analysis from this viewpoint refers to the operations to measure the fluctuation of physical parameters that can be expected in the real building.
- Addresses the developing construction-level information that expedites the generation of standard construction documents.

Researchers have been looking at extending BIM for many years to expand its scope. Below are some research on different domains: Lee et al. (2012) improved the design procedure by the efficient selection of alternative designs. They used Structural Building Information Modelling (S-BIM) to obtain more optimal solutions to improve eventually the constructability, structural safety, and economic feasibility of the building. Oti and Tizani (2015) employed current design process and data modelling techniques to model sustainability related information to inform decisions at the early stages of structural design. Chen and Hou (2011) developed a synchronous collaborative design platform for inter-disciplinary collaboration. This network platform allows for efficient multi-disciplinary collaboration in the development of BIM models. Suitable access rules and stable operating mechanisms are imposed in this work to maintain the integrity of the system.

Distribution of workflow efforts reflect the weight required for producing the project drawings. Demkin (2001) distributed the efforts for the traditional design process to be 15% for pre, conceptual, and preliminary design, 30% for design development, and 55% of construction documents. Very high efforts are required during preparing the detail drawings for the construction since many accumulated changes are made during the prior phases. Patrick MacLeamy (Figure 2.2) highlights the influence and impact of adopting the traditional and BIM process on the decisions and efforts of the completed building project (Smith, 2010). Curve 2 represents how the cost of design changes and Curve 1 how the effectives of those changes are varying during the timeline of a project from pre-design to operation.

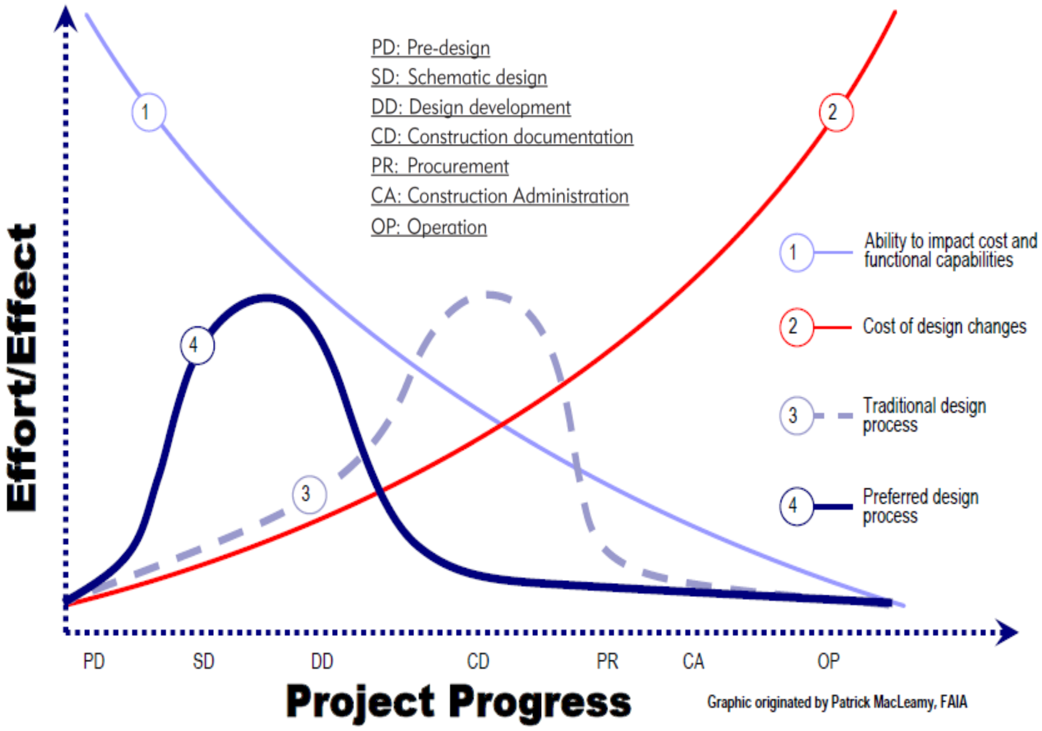


Figure 2.2 Relationship between design effort and time (Smith, 2010)

These curves illustrate the further effort of decisions made during the early stage of design process and how the cost of changes within the project lifetime is growing. Curve 3 shows the traditional distribution of the effort while Curve 4 shows the effort when adopting BIM. Within Curve 3, the peak of the effort coincides with a point in the project when the ability of the engineer to impact project performance is declining and the cost of making design changes is increasing and how the workflow effort Curve 4 can be restructured because of the adoption of BIM. The decision process is nearly completed before using BIM at the detailed phase when the ability to impact project performance is high and the cost of making design changes is low. MacLeamy developed the concept of “shifting the effort” to be between the schematic and developed phases to avoid encroaching into the detailed phase through using BIM process.

2.3.4 Cloud Computing and BIM

Computer network technologies have witnessed huge improvements and changes in the 1990s after which the use of the internet grew. The world began to see the power of distributed computing on a large scale. Most of the studies related to collaborative computing have focused on facilitating the coordination and communication aspects of collaboration. The evolution of computer-aided design started from standalone systems, to distributed and shared computing resources, and finally to a potential new paradigm, often referred to as cloud computing (Wu et al., 2015).

Through collaborative design, designers located at geographically dispersed locations share design data and assembly models to accomplish a common work. This type of collaboration usually needs the team members from different domains to access data/resources across enterprise boundaries. To achieve this collaboration, a well-managed system is required to maintain the consistency of data and transport such data between globally distributed sites (Wang et al., 2002).

The expression “Cloud Computing” has gained popularity during the last 10 years (Wong et al., 2014). It is real-time cloud-based collaboration that has provided new opportunities for data management in the AEC sector (Jiao et al., 2013). It is a significant advancement to address resources sharing based on business requirements and can be broadly defined as delivering hosted IT services over the Internet (Grilo and Jardim-Goncalves, 2011). It is a shared pool of easily usable and accessible virtualized resources that can be rapidly provided and released (Mell and Grance, 2009). There are some variations on what service is included within the Cloud Computing paradigm (Figure 2.3):

- Cloud Software as a Service (SaaS) - provides applications running on a cloud infrastructure. This means access to online software applications that are hosted by a vendor or service provider and are available via the Internet based upon time. Examples of SaaS include Rackspace Google Doc, Google app, and Microsoft Office Live.

- Cloud Platform as a Service (PaaS) - provides a platform and environment to allow developers to develop, run, and manage applications and services over the internet. This means that the users can develop their own applications and transfer them to other clients via the cloud. The advantage of PaaS is that it allows end-users to develop their own applications, libraries, and tools to support their services using programming languages. Examples of PaaS include Google App Engine, and Microsoft Azure.

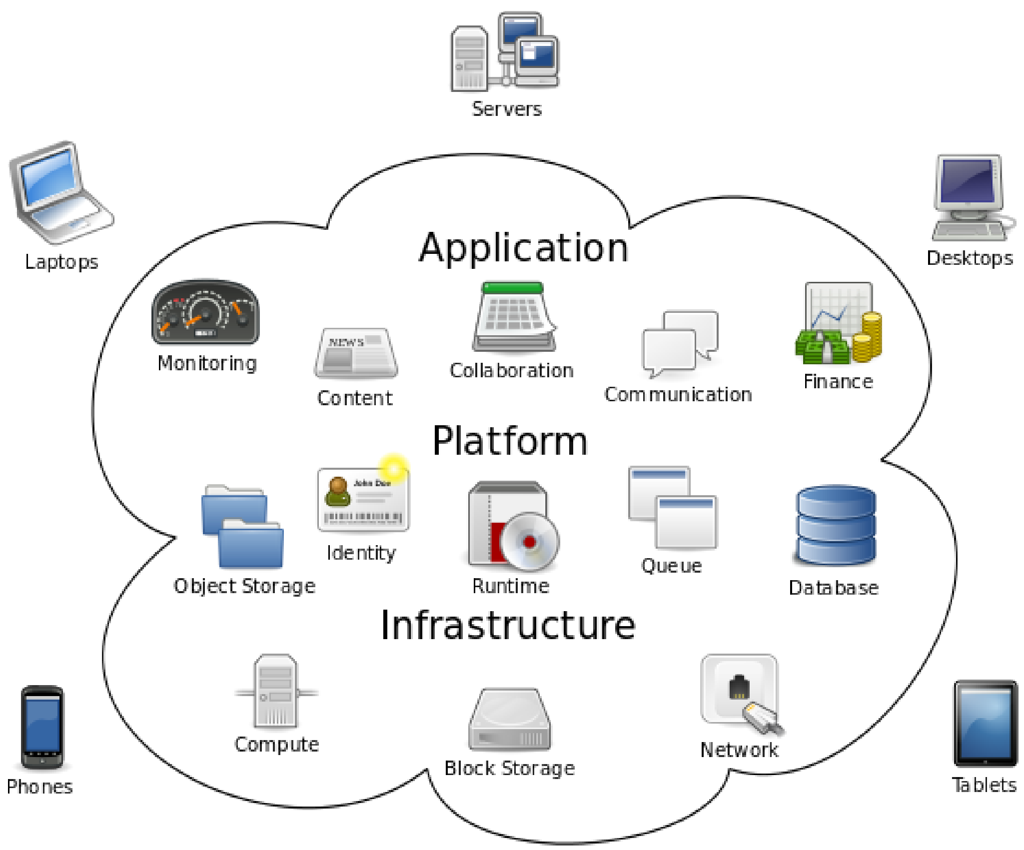


Figure 2.3 Services in Cloud Computing(Buyya et al., 2010).

- Cloud Infrastructure as a Service (IaaS) - provides hardware such as processing, storage, and network capacity, and other fundamental computing resources. IaaS can dramatically decrease the hardware cost for end-users. The user can build their own database on top of the infrastructure. Examples of IaaS include Amazon's EC2, Dropbox, Google Drive and Flexiscale.

As a result of the large and growing size of BIM project files and the difficulties inherent in managing model exchanges, there will be a growing demand for BIM servers (Eastman et al., 2011b). Cloud-BIM integration is a new generation of BIM development and is expected to produce another wave of change across the AEC industry (Wong et al., 2014). With distributed project teams, cloud computing provides designers with a centralized, computing platform for collaborative design using BIM. Amarnath et al. (2011) envisioned three distinct streams of applying BIM in the cloud. (1) BIM Model Server: a central BIM model of the building can be hosted using the cloud-computing platform. (2) BIM Software Server: current BIM software requires significant hardware resources to run. This hardware can be deployed in the cloud and shared efficiently between the project participants; and (3) BIM Content Management server: cloud computing provides a centralized and secure hosting environment for content in the form of data attributes/libraries needed for BIM usage and deployment.

The earliest publication on cloud-BIM was published around 2010/11 (Wong et al., 2014). Redmond et al. (2012) carried out interviews to discover how the information exchange process could be improved by the use of cloud BIM. They concluded that cloud-based BIM could create opportunities for different disciplines to share and exchange the essential data for making key decisions at the early design phase. Garg et al. (2013)

reviewed cloud computing technology and suggested a framework for the implementation of cloud computing for information sharing in the AEC industry. They found that interoperability is the key to the success of cloud implementation.

When BIM is deployed on a Cloud platform it further enhances the collaborative process that leverages on web-based BIM capabilities and traditional document management to improve coordination. There have been several recent reports on using cloud storage technology with BIM. Cloud storage is an extension of cloud computing, which collects, stores, and processes data based on services (Chuang et al., 2011). Cloud storage system enhances the real-time collaborative work between the BIM participants in the design phase and the construction phase (Amarnath et al., 2011). Ding and Xu (2014) constructed a BIM cloud storage system using cloud storage technique to combine the characteristics of BIM with the storage layer, infrastructure management, application interface layer, application service layer and access layer. Jiao et al. (2013) established a unified management-level cloud environment to support intra/inter organization data sharing and tracing, and automate data collection and correlation. In this work, as-built BIM project data in China was gathered and shared in the cloud to unified lifecycle data management. Rezgui and Director (2012) presented a prototype implementation of governance model by utilising the Cloud Computing paradigm. This prototype has been constructed using the open source CometCloud system and developed to allow data-transfer with third party tools. The initial work has been linked with Google Sketch up application to enable the integration of industry tools into the CloudBIM system.

2.3.5 Government BIM Strategy

Economic development in each society requires collaboration among all sectors. The UK construction industry is the sixth largest industry in the UK; it represents £ 110bn per annum of expenditure in 2010 and employing approximately 2.5 million workers (Cabinet-office, 2011). It has a substantial influence on UK economic development. There is a widespread acknowledgement across government that the UK does not get full value from the construction industry. With the momentum towards BIM increasing with time, the government and construction industry are collaborating closely to achieve structural change and to reorganize the way in which the construction industry is being managed. Based on reports by BCIS (2011), BIS (2011) and Cabinet-office (2011), the roadmap of a five-year strategy of the UK government is to restructure the construction sector outlook to enable the progressive use of collaborative BIM methodology on all government building programs by 2016, as well as providing a framework for exchanging information and delivery standards. The UK government strategy can become the catalyst to achieve the clear benefits that BIM promises.

BIM maturity levels have been planned to ensure clear articulation of the standards, the relationship with each other and how they can be applied to projects in the industry. It is being used at the moment at a number of different levels of sophistication and each level requires different capabilities of people, process, and technology (Masons, 2012, Langdon, 2012, Ngo, 2012), as shown in Figure 2.4.

- Level 0 - Use CAD in 2D with paper or electronic paper. The majority of design firms have been in this level for many years. Level 0 effectively means no collaboration.

- Level 1 – The increased use of CAD in both 2D and 3D format with start using BIM model in a simple format. Level 1 can be described as 'Lonely BIM' as models are not shared between project team members and each publishes and maintains its own data.
- Level 2 – This level uses BIM models by all key members of the integrated team. This level of BIM may utilize 4D construction sequencing and 5D cost information. The Government's BIM Strategy calls for the industry to achieve Level 2 BIM by 2016 (Ganah and John, 2015).
- Level 3 - is a completely open design process, it is fully integrated and collaborative process with the use of web services that are managed by a single, shared collaborative model and comply with emerging Industry Foundation Class (IFC) standards ((BIS), 2011, Langdon, 2012). It is expected that between 2016 and 2019 the UK Government and industry will move to Level 3 BIM (NBS, 2014) and the global construction market is forecast to grow by over 70% by 2025, which is deeply embedded in the wider digital economy (HM-Government, 2013).

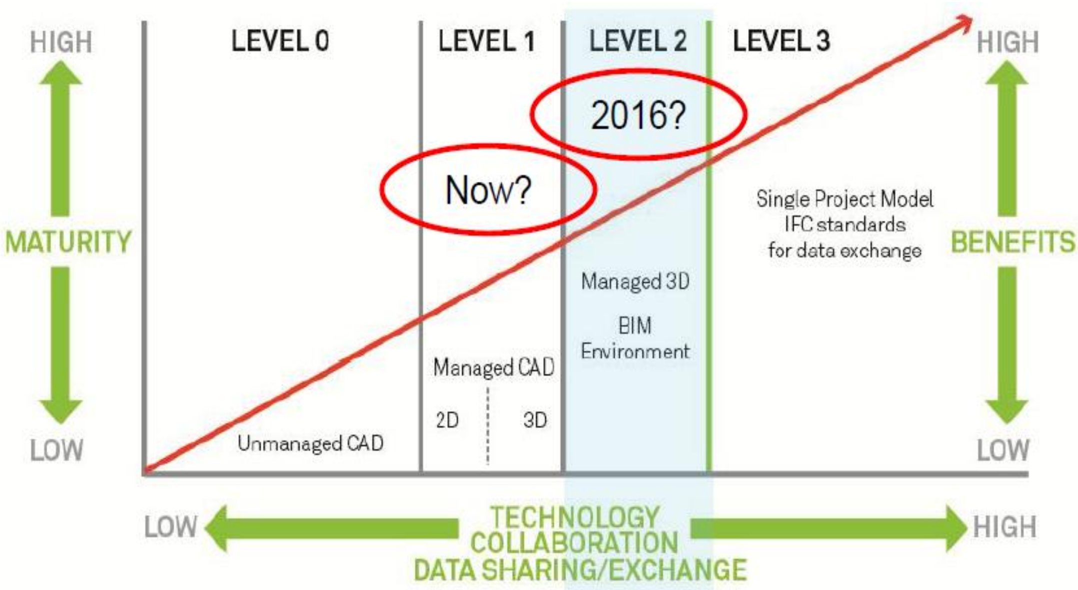


Figure 2.4 BIM maturity levels (Bews, 2012)

The concept of 'BIM levels' has become the accepted definition of what criteria are required to be deemed BIM-compatible. According to Bews (2012), only 15% of the projects in 2012 use level 2 BIM and the remaining are still at level 1 or less. The UK National BIM Report 2013 (NBS, 2013) showed that the percentage of UK engineering professionals using BIM on at least one project had jumped from 13% to 39% in the last two years. In the report of 2015 (NBS, 2015), the users that reached level 2 BIM requirements have grown to 59%, up from 51% the previous year.

Level 2 is the creation of a managed 3D environment with data attached, but created in separate discipline models. These separate models could be created with the client, architect, structural engineer, services engineering, contractor, and so on (Ganah and John, 2014). With vast amounts of digital data being created and shared during a project's lifecycle, a Common Data Environment (CDE) becomes an ideal environment in which to promote a collaborative working culture. CDE is a single source of information for the project. It is used to collect, manage and disseminate documentation, the graphical model, and non-graphical data for multidisciplinary teams in a managed process. It is simply a digital place, in which the information comes together. Creating this single place facilitates collaboration between project team members and helps avoiding duplication and mistakes (Deng and Di, 2013).

The CDE is a collaborative environment that is defined in PAS1192 and BS1192 documentation. These documents coordinate information with supply chain members on the project and outline what organisations need to do in the construction in order to reach BIM Level 2 compliance on their projects (Lea et al., 2015). The intent of initiating these documents is to provide a process for all of the parties involved with the projects to ensure that projects were delivered on time and under budget. The process is designed to cover both the information and the management processes and covers the project from inception to completion. It also covers the collaborative production of architectural, engineering and construction information (Maradza et al., 2014). Some of the benefits of a CDE have been summarized below (bsi, 2013):

- Reducing the time and effort required to check, version and reissue cycle.
- Extracting selections of the latest approved data from the shared area of the CDE.
- Reducing coordination checks, which are a by-product of the detailed design production process.
- Reusing of information to support construction planning, estimating, cost planning, facilities management, and many other downstream activities.
- Reducing the time and cost in producing coordinated information.

2.4 Change Management

Building design is a multi-disciplinary process involving contributions from an increasingly broad range of specialists. Managing changes on the models done by those specialists are a world-wide challenge that requires attention. The next sub-sections provide an overview of the changes in the design process and how to manage them. Then the integration of a versioning concept with design models is discussed. Finally, the evolution history of the versioned items of the model is represented.

2.4.1 Design change

Design changes are unavoidable due to the creative and iterative nature of design and specialist differences of the participants in the design process

(Gareis, 2010). The structure and the content of design are not static but subject to continuous changes from the preliminary stage of the design to even after construction has started (Love et al., 2011).

Changes are mostly caused by the disciplines involved in the design, construction, manufacturing and management process of the project (Hao et al., 2008). It might come from the client desire for changing some current information in the design model, architect view for getting new design ideas, structural designer requirement for improving the stability of the building, fabrication request for changing unavailable specification and so on for the other disciplines involved in the project (Motawa et al., 2007).

Changes in AEC projects are common and likely to happen at any stage of the project, from different sources, by various reasons, and can have significant effects. During the design and construction of the project, many decisions often have to be made based on incomplete information, assumptions, and the personal experiences (Shourangiz et al., 2011). Design changes stem from different reasons that can be summarized into three kinds:

1. Design errors: refer to re-doing a process or an activity that was incorrectly implemented in the first time. This type is more related to the experience of the designers.

2. Design requirements: refer to re-doing a process or activity that was done according to some design demand and necessities required for some designers.

3. Change Order: refer to changes generated by some sources; usually they are outside the boundaries of design teams. For instance, scope changes from the owner, cost reduction or extension that demands redesign of the conceptual model.

Table 2.2, which was combined from Pilehchian et al. (2015) and Lock and Flouris (2012), highlights a number of important characteristics that are significant for classifying changes in any design project. This table lists the type of changes, the type of component features that can be changed, the dependencies between components, the degree of effect of the change on the other designers, the dependencies of the changes on the design teams, the time of the change during the project lifecycle, and the change effect on the time and cost of the project.

Table 2.2 Different characteristics for classifying changes

Key characteristics of changes	**Sub-characteristics of changes**	**Description/ Example**
Change Type	Addition	Creating a new component
	Modification	Modification in one or several features of an existing component
	Deletion	Deleting an existing component
Changed Component features	Geometry	Shape: cubic, cylindrical, rectangular, plate
		Dimensions: length, width, thickness, diameter, slope
	Position	Coordinates: X ,Y ,Z
		Orientation: Rx, Ry, Rz
	Specification	Material: concrete, mild steel, galvanized steel
		Elements: Stud, Rebar: size, shape, arrangement
		Semantic Properties: Fire-rating, acoustic, water proof
Dependencies between Components	Spatial	Connected To : column and floors, main and secondary ducts
		Adjacent To: duct and adjacent pipes, duct and ceiling
		Supported By: duct and steel hangers
		Surrounded By: duct and false ceiling/ plenum area
	Analytical	Structural Integrity: size of sleeves and arrangement of rebar
		Architectural Consistency: functionality of room and exposed duct

		Mechanical Interaction: location of air supply duct
		Electrical Relationship: size of cable tray and motor power
		Operational Requirement: clearance around a pipe
Level of Change	Conceptual	Change in basic documents, design, specification with fundamental effect on many designers
	Primary	Major change in main components position, geometry, etc., which affect other designers
	Secondary	Minor change in component elements or features with minimum effect on other designers
	Non	change in component elements or features with no effect on other designers
Type of dependencies	Intra- model	Change affect components of the same disciplinary team
	Inter- model	Change affect components of the other disciplinary team
Change Timing	Conceptual design	During early decision making about the primary aspects of design
	Basic design	During early stages of the design but prior to the full extended design
	Detail design	During the extended design but prior to any procurement /construction
	Procurement	After Purchase Order but prior to fabrication
	Fabrication	After Fabrication but prior to erection
	Construction	After commence of construction
Change Impacts	Cost impacts	Major: considerable effects on costs
		Minor: insignificant effects on costs
	Time impacts	Major: considerable effects on schedule
		Minor: insignificant effects on schedule

Changes often result in unanticipated side effects in time delays, cost overruns, and quality defects (Lock and Flouris, 2012). The impacts of changes are sometimes overlooked or revealed later at the end of the design process or even in the construction process when later handling with the changed items or implementing them in the reality (Isaac and Navon, 2013). One study estimated that 78% of quality problems can be traced to design and 40 to 50% of the total work hours invested by the designers of the project have been estimated to be related to design changes (Koskela, 1992). Another study estimated that between 20 to 25% of the total construction period is lost due to deficiencies in the design (Alarcón and Mardones, 1998). One of the detrimental consequences of project changes is revision or rework of the design. Sun and Meng (2009) estimated that the cost of rework of the design in construction projects can be up to 10–15% of the contract value. Consequently, successful management of design changes is of vital importance for the efficient delivery of construction projects.

2.4.2 Managing Changes

Building design and construction is a very complex process. The amount of available information in early design stages increases with time in terms of quality and details of information. Even with careful control of the process, it is inevitably prone to numerous changes made and revisited decisions at various times of the project lifecycle. Any change made by any designer may affect other disciplinary designers. For example, the structural engineer, needs to know when other teams make changes to their model since that may affect his model, and vice versa. Since changes cannot be avoided in any construction project, the requirements for change management become necessary for all aspects in the life cycle of the building (Holzer, 2015).

Effective management of design changes provide a collaborative advantage in the AEC industry through identifying, planning, implementing, documenting and evaluating changes to design models. It has two main goals: supporting the process of changes and enabling traceability of changes in the design process (Crnkovic et al., 2003). Change management system has been identified as an important area of the building project (Aslani et al., 2009, Turner, 2014). Ibbs et al. (2001) introduced project change management system to minimize deleterious and promote beneficial changes. The developed system is founded on four principles: promote a balanced change culture, recognize, evaluate, and implement change, and continuous improvement from lessons learned. Shen et al. (2010) classified managing changes of the model to get a collaborative environment into five stages in sequence: identification, evaluation, proposal, approval, and implementation. He pointed out that the most costly works often relate to design changes and design errors. A new system is proposed by Motawa et al. (2006) to evaluate the effects of changes in construction projects. The system simulated the relations between the causes and effects of changes, and intended to facilitate proactive change management in projects.

A database management system (DBMS) is used for managing data. It makes possible for end users to create, read, update, retrieve, delete, and manage data in a database (Coronel and Morris, 2016). Several research efforts in the use of information technology for managing the design change in building projects are investigated. Mokhtar et al. (1998) presented an information model to help the coordination of design information through managing design changes. A central database is developed in this study to store the building components data and to make these components active in assisting the coordination process. Nour (2012) addressed the problem of creating a BIM workspace for the virtual organization, that consists of several stakeholders. He proposed an information management system at the object level for dependency management changes and access rights allocation. Cloud computing technology offers an effective way for different design teams to share building models. Sawhney and Maheswari (2013) proposed cloud based framework to allow access to a central server which serves as a host for numerous BIM software packages. Many local servers are identified for each disciplinary team for developing their design models. If a designer decides to make the changes on some objects, these objects will be locked in the central and other local models. Then, the changes are recorded on the objects and the objects are unlocked as soon the designer save his model. Isaac and Navon (2009) developed a model as a basis for change control. This model identified and focused attention on the possible impact of proposed changes in building projects before these changes are implemented in the project design. The model uses available sources of project information to detect the impact of changes on cost, schedule, and performance of the project.

Other research efforts have investigated using realistic cases that had a better simulation of real-project conditions to deal with change management. Akcamete et al. (2009) performed two case studies and observed work orders from daily maintenance to understand the types of changes that occur throughout the project life cycle. They discussed some challenges related with managing changes and updating BIM models accordingly, as well as how well commercially available systems address these challenges. This analysis emphasizes a need for computational support for change management. Pilehchian et al. (2015) described an approach to represent, coordinate, and track changes within a collaborative multi-disciplinary BIM environment. A case study of a BIM project was used to investigate numerous design changes. This approach characterizes design changes in ontology to represent change impacts in a BIM-based project delivery process.

2.4.3 Versioning

The maturity and the evolution of the building elements of a model are related to the time and the information availability. While these building elements in the current model reflect a final state, they may also be continually revised over the course of a project lifecycle (Peterson et al., 2011, Jaly-Zada et al., 2014). With the purpose of, ensuring consistency of distribution models, dealing with evolving information, easing of tracking changes, and supporting of retracing change history, building models have to be versioned (Koch and Firmenich, 2011, Kim et al., 2011). It is a solution of the design change management. Therefore, due to the parallel and iterative nature of the design process, several model versions can be created and distributed between multi-disciplinary designers. Each model version is created by applying a series of processes on a predecessor model version. The series of processes may include the creation, elimination, and modification of different information that constitute the predecessor model version (Roldán et al., 2010). Changes to the model are presented by incrementing an associated number or a letter code, termed the "version number" or simply "revision" or "version" (Milentijevic et al., 2008).

Versioning has long been in the focus of scientific researchers for various purposes. Many researchers (Roddick, 1995, Conradi and Westfechtel, 1998, Gyssens et al., 1994) formulated the specifications and requirements of versioning, which represent the early state of theoretical concepts to this approach. Versioning moves even further towards the implementation of the theoretical concepts using engineering applications. Document Management Systems (DMS) are used to manage at the document level, using computer programs to keep track the different documents that are created by different users (Kitagawa et al., 1984, Wittenburg and Little, 1994). It is used in the design process to manage the project at the model level and to create a linear history (link) for the models. This system for managing document/model does not appropriately support the collaborative process because it does not deal with or compare the contents of the files (Firmenich et al., 2005).

More advanced, many companies consider product data management system (PDMS) of software to track and control data related to a particular product (Peltonen et al., 1996). PDM systems are used to control information, files, documents, and work processes required to design, build, support, distribute, and maintain products (Liu and Xu, 2001).

Version control systems (VCS) have been adopted by many researchers to manage changes of text- based documents, computer programs, web sites, and other collections of information as an aspect of software configuration management (SCM) (Spinellis, 2005). For almost four decades, VCSs are an crucial part of the infrastructure necessary for an effective software development process (Altmanninger et al., 2009). the VCS is essential in merging and branching, integrating with tracking issues, returning back to an old revision (Vesperman, 2006). This system is based on comparing text files through looking for longest common sequence of characters to find the text changes. It does not know the syntax of the file and understanding changes (Koegel et al., 2010). VCS runs as stand-alone applications or embeds in various types of software (Firmenich et al., 2005).

Hass (2003) used VCS approach to managing and versioning changes of the source code in the software program. VCS is used with design models on the basis of object versions as the smallest entities “object VCS” (Richter and Beucke, 2008). Each text file represents as object, the system then can find, for instance, that there is a new object because there is a new text file. Firmenich (2005) proposed modelling a single object version instead of a completely building model. A central data store “repository” had been suggested by Beucke (2006) to collect and study individual results of different model versions. He mentioned that the separate and isolated version information could be stored in private workspaces for supporting collaboration. Nour et al. (2006) extended the idea to include data models. He proposed a database for tagging versions of the design model using object line number identifiers and IFC GUID. A sub versioning approach has been used by Pilato et al. (2008) through adding an extra repository with the central one to synchronize different models. Wang et al. (2007) presented a semi-automated approach for matching and identifying differences between two models and updating the existing matches using upgrade patterns developed based on taxonomies of version differences. Gonnet et al. (2007) developed a Collaborative Model for representing different design states (CoMoDe) by capturing design object versions that arise during a design process.

Nour and Beucke (2010) clarified that since building designs are represented in the form of objects and DMS does not support object-level management, information management systems could be proposed at the object level through versioning model’s objects and managing changes in the objects. Koch and Firmenich (2011) proposed a new processing-oriented building model through integrating the existing state-oriented descriptions with the additional change-oriented information, as modelling operations. The proposed model therefore contains both the design states and the design changes. The changes were recorded utilizing a new operative modelling language. This approach introduces new concepts for distributed collaboration and model management. Richter and Beucke

(2006) presented different solution approaches for the “diff” and “merge” of documents getting benefit from versioning sets on a central server and comparing objects with the same persistent identifier (POID). Koch and Firmenich (2006) described a novel “diff” and “merge” approach based on operative models. The applied operations describe the differences between two versions of the model instance because the semantics of differences are explicitly stored.

The key to well-structured models are a Common Data Environment (CDE); an online place for collecting, managing and sharing information amongst a team working on a project. The CDE could take many forms, depending on the size or type of the project. It could be a project server, an extranet, or a cloud-based system (CDP, 2014). CDE provides not just a secure central repository for all project information but provides version control and user security for all data and files. (bsi, 2013). This gives users confidence that they have access to the very latest version of information and provides those tasked with controlling information flow with an oversight of who has seen and done what and when. This accountability and transparency combined with the ease of information share provided by a cloud based CDE allows for more collaborative working whilst maintaining control (Payne, 2013).

2.4.4 Version graph

A version graph is the typical graphic representation of versioning. It mainly represents the evolution history of a versioned item (Kim et al., 2011). The graph consists of a set of versions (represented as nodes) connected by relationships called successor relationships (represented as arrows pointing from older to newer). The new versions are based on past revisions, For instance, if V_1 and V_2 are two versions, then V_2 is a successor of V_1, this means that V_2 has been derived from V_1, such as by modifying a copy of V_1. Version graphs may have different shapes. Conradi and Westfechtel (1998) classified version graph into three types, Figure 2.5:

- **Sequence graph**: this is the simplest case; each version is based on its immediate predecessor alone, and they form a simple line with a single latest version without any branching. This type is more common in the building design process when the designers issuing multiple sequential model versions with the time.
- **Tree graph**: this type can be divided into smaller branches. Different new versions are created based on a past version. Therefore, each node can have more than one child. It can represent this type in the building design process when two or more designers are working on different parts of the model that there is no relation between them.
- **Acyclic graph**: this is one of the most complex aspects of revision control. It is a graph with no cycles, it is similar to the tree graph but the past versions in different branches are merged into a single new version that incorporating the both past versions. Therefore, many nodes have only one child. In the presence of merges, the resulting graph is no longer a tree. This type is now widely used in the building design process due to technological development. Two or more designers that are working on different parts of the model (even with the whole model) can merge these parts to create a single model version at the end.

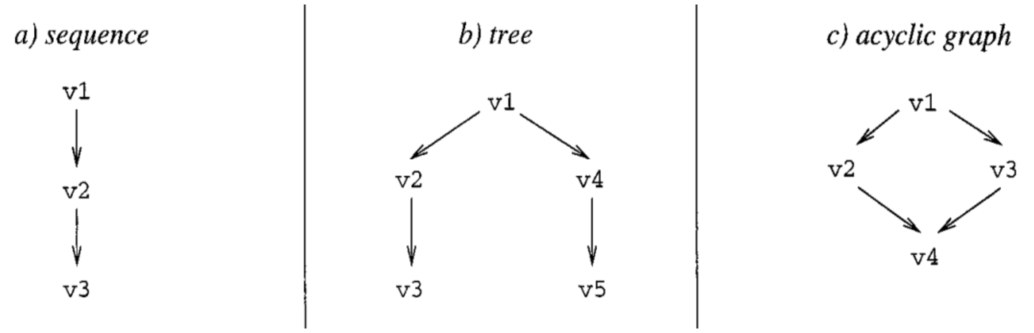


Figure 2.5 Version graphs (sequence, tree and acyclic graph)(Conradi and Westfechtel, 1998)

Many scientific researchers (Cellary and Jomier, 1990, Firmenich, 2004, Richter and Beucke, 2008, Taentzer et al., 2014) have used the versioning graph for representing model versioning. Nour and Beucke (2010) provided a versioning graph for the model and object versions of the data model. Kim et al. (2011) used version graph to illustrate a typical design versioning schema through considering alternative design versions. Koch and Firmenich (2011) utilized version graphs to describe model versions, model changes and model processing.

2.5 Challenges of Integrating BIM in the Design Process

The contemporary building design process is progressing from traditional CAD systems towards the adoption of BIM in the projects development process. This comes with challenges in information management as BIM is yet to be fully matured (NIBS, 2007). The technology to collaborate on BIM has not yet delivered on the AEC industry requirements. Deutsch (2011) presented twelve different obstacles to successful adoption of collaboration in BIM and integrated design (Figure 2.6). The following section discusses the related challenges in this study when integrating BIM in the design process to deal with some of the aspects on Figure 2.6.

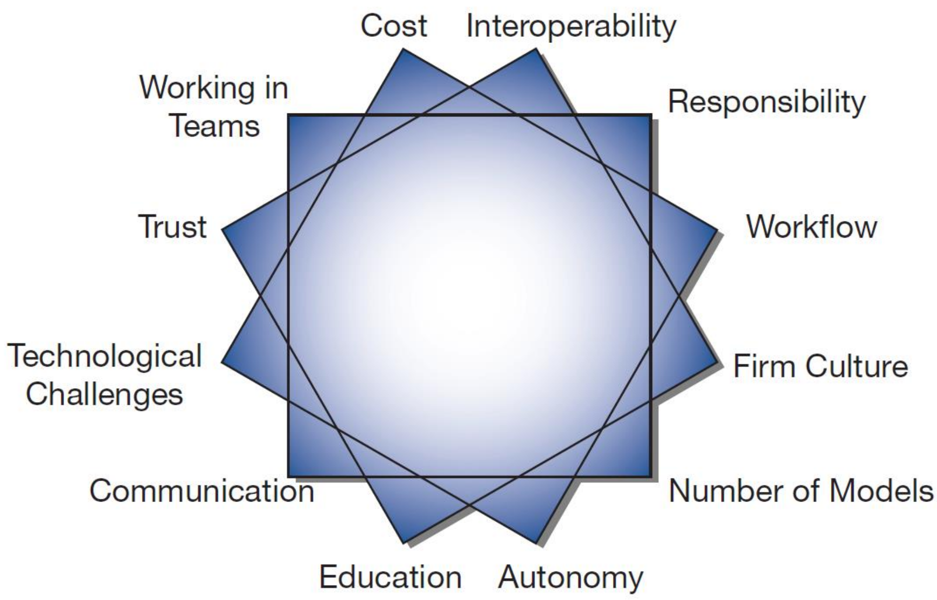


Figure 2.6 Obstacles to successful collaboration in BIM and design (Deutsch, 2011).

2.5.1 Multi-Disciplinary Teams

Building design is a comprehensive and continuous iterative process, which includes a wide range of issues that require many technical details and multiple engineering expertise (Clarkson and Eckert, 2004). The engineering backgrounds of each disciplinary designer who has participated in any project are different. Structural designers, for example, deal with the design to keep the structure safe, serviceable, and economical, whereas architects seek to control the aesthetic realistic and practical considerations, and so on for the other disciplinary designers (electrical, HVAC, sanitary engineers, etc...).

Each discipline is responsible for its contribution to the overall design, all focusing on their own work, which leads to coordination problems. Engineering projects depend, to a large degree, on effective collaboration and communication among separate disciplinary teams (Orviz, 2010). Collaboration among multi-disciplinary designers, during all design stages, has a crucial impact on the success of any engineering project. Integrated design can be achieved when all actors of the project collaborate across various disciplines and agree on far-reaching decisions jointly from conception to completion (Ruikar et al., 2005).

Many researchers suggested improving the collaboration among the multi-disciplinary teams. Geryville et al. (2007) described a collaborative framework system to exchange and share information. He mentioned that a structured team could facilitate information sharing, and exchanging; and consequently contribute to the success of the design project. However, the collaboration among the multi-disciplinary teams is not only to ease sharing and exchanging the information among them but also to truck the changes in this information and to clarify them clearly to the other disciplinary teams.

2.5.2 Different BIM Models

Multi-disciplinary designers tend to work with different BIM models depending on the speciality of each one of them. The content of BIM models is not typically the same among different designers. The structural BIM model, for example, combines a physical representation fully associated with an analytical representation of the building. Many researchers used the concept of working in one central BIM that is shared among all disciplines (Ashcraft, 2009, Isikdag, 2012, Nour, 2012). Nevertheless, all partners in the project work in the same central BIM model have many difficulties related to manage both the shared as well as the specific information for the different disciplinary models.

Designers are dealing with a wide and a common area of information that each is looking at it according to his/her engineering point of view. There are some pieces of information in BIM shared among different designers that each can use them on its BIM model several times. For example, beams, columns, slabs, and walls are not only essential elements for structural designer, but also they are significant for other disciplinary teams. Changing the features of these elements (such as length, material type, locations, etc.…) by any designer must be transferred to the others for approval (Macdonald, 2013). On the other hand, steel bars, cooling ducts, furniture, etc.… are specific elements of one type of disciplinary designers that only this team is affected with their changes. Isaac and Navon (2009) developed a model as a basis for change control. This model identified and focused the attention on the possible impact of the building projects on the cost, schedule, and performance of the project before these changes are implemented in the project design and transferred to the others. However, this information by one of the disciplines may or may not have an impact on the other disciplinary teams. Therefore, classifying information in terms of their impact on the other disciplinary designers and clarifying the affected information to them have a significant influence on the management of building projects.

The interrelated information is needed to manage properly to control the ownership and ensure the consistency of the information among all BIM models. Therefore, shared information in BIM models requires to be managed among inter-disciplinary designers while specific information requires to be managed among the related intra-disciplinary designers. Chen and Hou (2014) proposed an internet-based multi-disciplinary modeling collaboration to integrate design data from multiple design teams. The proposed system is based on a hybrid client–server and Peer-to-peer “P2P” network for supporting both inter-disciplinary and intra-disciplinary modelling collaboration. Through the proposed mechanisms, replicated modeling collaboration was achieved through using P2P network "local server" across each intra- disciplinary teams, and centralized modeling collaboration was accomplished through using client-server network "global server" within an inter-disciplinary team.

In the centralized model, the global server provides teams with a central BIM model and sends this stored BIM model to each team's local server. This means that there are several central models shared among the intra-disciplinary teams. While in the replicated model, it allows connecting a group of computers with equivalent capabilities as well as responsibilities to work together. From the perspective of the author, a single central model that is shared among the whole disciplinary teams can easily ensure data consistency as there is only one master copy saved in the central database. Moreover, the decentralized connection is not suitable for a large group of users to pool resources due to the restrictions of the capability and the complexity of maintaining data consistency among BIM models.

2.5.3 Different BIM Applications

The very fundamental idea for integrating and collaborating two or more BIM systems or applications is to enable them to communicate, share and exchange information (Tizani, 2011). Engineering design is a complicated activity that not only many designers are involved and various models are used in this process but also a large amount of data transfer among different applications and between each of the design stages. These different applications usually use their own proprietary BIM models to store information. Different BIM software packages are used in the AEC industry (such as, Autodesk Revit, ArchiCAD, AECOsim Building Designer, CodeBook, DDS-CAD, Digital Project, Nosyko dRofus, OpenStudio, Synchro PRO, Tekla Structures, Tekla BIMsight, Vico Office, Navisworks, MicroStation, VectorWorks Architect, Allplan, GRAITEC Advance, IDEA

Architectural, FINE MEP, VisualARQ). These BIM software that are used at each stage of the design process do not necessarily allow for the properly reading the exchanged models due to the incompatibility among them (Bruce A. Burt, December 2009). This generates a high level of re-working on data. Such data and information flow leads to a lack of integration between different stages in the design processes and hampers collaborative design (Fahdah and Tizani, 2008).

This information must be readable by other applications to achieve the interoperability among different BIMs (Nizam and Zhang, 2015). The first step in the interoperability process is to translate the information from the native BIM model to a common, universal, and standard data model, to be able to transfer it between different systems and applications, and including translating back into the native BIM model of that application. Interoperability of heterogeneous applications can be best achieved by using generalized and standardized representations of the BIM model to describe BIM as a data exchange standard. The most widespread exchange formats is IFC (Industry Foundation Classes) standard. It is used by most of the BIM applications. However, the IFC standard does not support the whole domains and scopes in the AEC industry and more developments are required. That is why; researchers and vendors have urged to advance the capability of IFC for a long time.

For instance, Ma et al. (2015) proposed extending the IFC schema to represent the damage modes of RC structures. Cemesova (2013) suggested new property sets to support the design of low energy buildings and so on for the other AEC domain. However, the adoption and implementation of the IFC standard for managing the design changes in BIM models was not addressed and, in the authors' opinion, remains a challenge. The interoperability of BIM and the IFC models will be discussed in more details in the next chapter.

2.5.4 Change Management

The models that are used by different disciplines are large, complex, and highly interdependent (Steel et al., 2012a). The structure of design information is not static but exposed to frequent changes. Changes in different design models are very common in the AEC industry. It is not practical to circulate each single change that occurred or solution found straightaway to all others involved in the project. Instead, each designer usually requires longer time to work with his private model separately to develop and increase the maturity of the model and try to find acceptable solutions before distributing the result to others.

The information in any new release of the model version does not necessarily affect all the participants in the project. For instance, if the latest information is about changing the distribution of the reinforcement bars of some slabs in the model, then this information is essential for the structural team only. While changing the location of some columns are important for all designers' teams. The contents of the design changes in each of the BIM models must be clarified and transferred clearly to the affected disciplines to make sure that all are working on the latest revision of the model (Macdonald, 2013). Mistakes begin to creep into the works when updates are incorrectly done or incompletely prepared, and work will be wasted since designers are working on the out-of-date information.

Managing those changes is a core concern of any design process to ensure that all changes are assessed and reviewed by the right person and in a controlled manner. Model versioning is a way to support change management at the model level that can be implemented at the very early design stage to improve the collaborative design. Firmenich (2005) proposed modelling a single object version instead of a completely building model. Therefore, collaboration among the engineers was obtained by version management on an object basis of the project data. However, in design projects, each building element (for instance, beams, columns) has a set of information "features" that make up this element (for instance, shape, location, etc.…). Version management of a single element does not mean that all the element features are changed. Therefore, in the authors' opinion, versioning the information in the project can be classified into three gradual levels: versioning the whole building model, the element, and then the feature. Through this distribution, the changes in the project can be classified granularly to cover the smallest changes that affecting the designers.

The current BIM tools provide limited support in managing changes of several disciplinary models. Each discipline needs to compare the current model with the preceding model to identify the latest changes in the elements and features that might affect their work. At the same time, tracking the history changes on the different disciplinary models requires all previous model versions to be stored in a secure place and each discipline needs to do the comparison process between the whole model versions to identify the required information (Eastman et al., 2011a). A model that holds the current information with the changes information of the whole disciplinary models has not been explored yet. This extended model can reduce the process of finding the design changes into one designer who generates the latest model version and can reduce the number of the model version files into a single file that stores the current with the history information associated with different BIM models.

2.5.5 Shared storage

The designers' teams from different domains that are at geographically dispersed locations need to distribute and share the new release of the BIM model in a centralized shared storage. Deploying the BIM in a shared storage "like cloud" can further enhance the project integration. This shared storage can connect a large number of disciplines through a real-time collaboration among them.

Managing access to the shared workspace is particularly important to guarantee that the right participants in the right circumstances are using and dealing with the new BIM release. Defining roles-based users' access to clarify the permissions to the authorized users to access to the certain workspace is essential to establish collaboration practices in the AEC industry. Roshani and Tizani (2005) proposed a COllaborative Design Environment "CODE" system via web to integrate the design of distributed project participants. The shared system allows different actors to connect and address problems together, and exchange ideas and information. The process of selecting the actors and identifying the discussion issues have to be managed manually. Moreover, there are no role-based access rights to manage the permissions to contact with the related disciplines. Therefore, the fundamental barriers related to which designers have the right to access and use information remains.

(Nour, 2009b) suggested the use of a private repository that enables the stakeholders to keep their unshared information within the boundaries of their organizations and only exchange their own local private domain data among themselves. This enables to use any type of software or developing platform and use a data repository in a homogeneous software environment. At certain development stages of the design, a release version can be uploaded to the central project's server to be communicated to other domains. However, this process needs to store the required information with the related database. Thus, it needs to be highly efficient to manage the information in the database in parallel with the different versions of the exchange models or partial models.

2.6 Summary

This chapter reviewed collaboration in the building design process and improved the process of adoption BIM technology. The chapter then provided an overview of the changes to the building models during the design process, and improved the coordination and organization of the design changes by using the versioning concept.

The fragmented nature of the design process is a main obstacle to the efficiency and integration of the design process as a whole. The current collaborative design process is typically expensive and time-consuming because of the inconsistent management of design changes can result in many disruptive effects. Some of the main challenges of integrating BIM in the design process are discussed in this chapter that can be summarized below:

- Multi-disciplinary designers are involved in the design project have different competency and responsibility.
- Multi-BIM models are used among different disciplines make managing them difficult.
- Multiple and incompatible BIM applications make reading and dealing with BIM models difficult.
- Weakness in the management of changes in BIM models leads to inconsistencies between disciplines.
- BIM models include shared and specific information that changing them affect all/some/or none of the other disciplines.

- Current versioning process target BIM models as a whole without taking into consideration versioning the elements and features information.
- Comparing two sequential model versions by each discipline to determine the design changes consume a lot of time by every designer.
- Current process to study the change history of the building elements is neither easy nor accurate.
- Access to shared storage needs to be managed better to ensure effective collaboration among designers.

Chapter 3

Industry Foundation Classes for Interoperability

3.1 Introduction

This chapter presents the review on BIM Interoperability (Section 3.2) and Industry Foundation Classes (IFC) (Section 3.3). The assessment on IFC specification has been expanded to describe the data schema (IFC standard) and the data file format. The chapter examines different research papers, standards, and reports about extensions of the IFC schema and uses of the IFC file (Section 3.4). Overviews of the main applications that are dealing with the design changes are provided in (Section 3.5). It concludes with a discussion some of the main limitations in using the IFC for the management the design changes (Section 3.6).

3.2 BIM Interoperability

Like many other industrial sectors, a major challenge faced by the AEC industry is the lack of adequate level of interoperability among BIM-enabled software applications (Karan and Irizarry, 2015). SmartMarket report highlighted this challenge among applications as top of the list that need to be addressed to maximize the benefits of BIM (McGraw-Hill, 2015). The participation of multi-disciplinary teams in engineering projects (including owners, architects, engineers, consultants, contractors, suppliers, etc.…) and the distinct nature of the various stages of the project lifecycle have led to the use of different and inconsistent software applications. It is not necessary for a BIM application to allow the reading of different BIM models created by the other applications due to the incompatibility among them. Interoperability between BIM tools is limited, as they have been developed in isolation and have different internal rules applied to the models. This demands a high level of re-working on information and leads to a lack of integration between the different stages in the design processes and hamper collaborative design (Oti et al., 2014). Sun and Aouad (2000) argued that the objective of interoperability is to achieve coherent management and electronic exchanging of information and knowledge during the construction of projects. Grilo and Jardim-Goncalves (2010) illustrates some main problems on interoperability of systems:

- It is not easy to access accurate information and data in an appropriate time at any stage of the project Lifecycle.
- There is a shortage of compatibility between systems. A shared system for managing projects information does not exist.
- Improvement tools and programs for design and management optimize for a limited range of factors in limited domains.

- Collaboration issues have not given priority in the project and do not effectively consider all Lifecycle issues.
- Modelling and planning do not effectively take all aspects of the building Lifecycle into consideration.

Therefore, the integration of systems has become a significant quest to achieve efficient and effective collaboration and interoperation (Gallaher et al., 2004). In reality, systems integration and collaboration are all about interoperability (Shen et al., 2010), which is constitutes the ability to exchange information and make it useful (Oxford.Dictionaries, 2012). To be more specific, it refers to the ability of two or more separate systems or software programs to manage, communicate, and exchange data with each other. Steel et al. (2012) divided interoperability into four levels:

- File level interoperability is the ability of two tools to successfully exchange files.
- Syntax-level interoperability is the ability of two tools to successfully analyze files without mistakes.
- Visualization-level interoperability is the ability of two tools to efficaciously visualize an exchanged model.
- Semantic-level interoperability is the ability of two tools to come to a shared understanding of the meaning of a model being exchanged.

The issues concerning interoperability are twofold. Firstly, data exchanges need to be standardised in order for different application tools to communicate using the same concepts. Secondly, the data that is transferred may need to be processed in order to become useful for the target application. Shen et al. (2010) illustrated systems interoperability from two different viewpoints: (1) frameworks interoperability, depends on common communication languages and protocols and (2) data interoperability, focuses on common data models or formats.

- **Framework Interoperability**

When two software systems need to work together, they communicate based on agreed standards. Framework interoperability is an overarching set of policies, standards and communication languages, which describe the way in which organizations have agreed to manage different BIM applications with each other. Therefore, an interoperability framework is not a static document and may have to be adapted over time with the technologies and standards requirements changed.

- **Data Interoperability**

Data interoperability is the ease with which data generated by any software can be correctly understood and interpreted by the others. The enabling technology for data interoperability is data modelling. In heterogeneous applications, sharing data requires everyone to have a neutral and common data model (Steel et al., 2012b). With this data model, it is possible for building information to be developed and used again in the rest of building lifecycle and this is the most feasible solution in the AEC industry (Motamedi et al., 2016). To exchange different BIM models and achieve an interoperable environment, source BIM application must implement a compatible data modelling language and generate a neutral data model that represent the source BIM information, as illustrated in Figure 3.1. Other target BIM applications that receive the neutral exchange file have to use the same language for accessing the neutral file, interpreting its contents, and creating an internal representation of that information (Bakis et al., 2007).

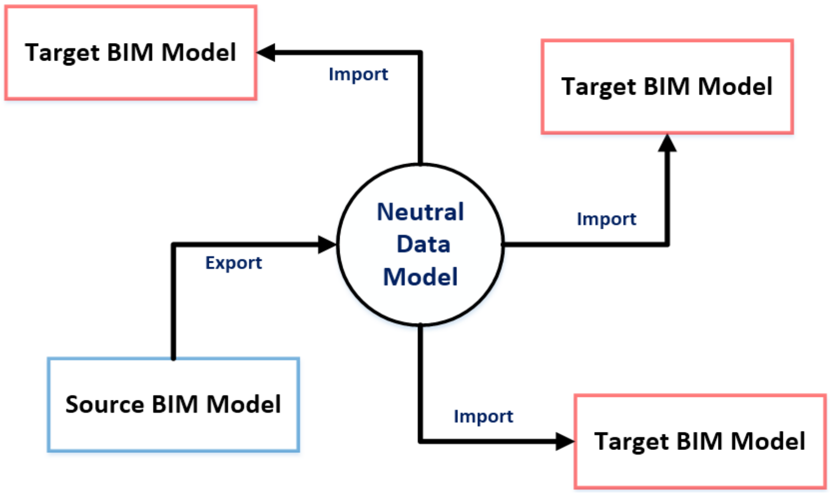


Figure 3.1 Importing and exporting neutral data model

3.2.1 Data Exchange Standardisation

Standards have played and will continue to play important roles in the AEC environment. Some low-level methods to exchange data with other applications have been developed since the late 1970s. The first attempt to develop building model standards for file-based exchange formats was in the 1980s. This early works were limited in the exchange of geometrical information between CAD applications, such as Drawing eXchange Format (DXF) and Initial Graphics Exchange Specification (IGES) (US.PRO, 2006, Eastman et al., 2011b). With the increasing need to transfer more semantic information, data models developed from the mid-1980s to support the product and object model exchanges within different industries. These standards were all brought together under the International Standard Organization (ISO); and the Standard for Exchange of Product data (STEP) was developed to define not only standard data models to facilitate information exchange, but also to define a standard methodology for data modelling and data exchanging (Amar, et al., 2000). ISO-STEP developed EXPRESS data model language to be a basis for the electronic exchange of product data between computer-based product life-cycle systems (such as Industry Foundation Classes “IFC” and

CIMSteel integration "CIS/2") (Eastman and Augenbroe, 1998). Apart from EXPRESS, another huge set of exchanges are supported by XML (eXtensible Mark-up Language). It is an extension to HTML, the base language of the Web (Bosak and Bray, 1999). Some other standardisation efforts focus on specific subjects on data exchange for the publication of a subset of model information concerning delivering building information on geometric modelling (such as the Construction Operations Building Information Exchange (COBie)). Generally, Industry Foundation Class (IFC) and Green Building XML (gbXML) are two prevalent information infrastructures in the AEC industry that are used for common data exchange between AEC applications (Dong et al., 2007). To examine further the contributions of relevant standards to interoperability, ISO-STEP standards are discussed.

3.2.2 ISO- STEP

Since 1984, ISO has been working on the development of a comprehensive standard for the exchange of product data between computer-based product life-cycle systems (Geiger et al., 2015). Initial awareness has been on design and manufacturing applications. ISO-10303 standard (informally known as STEP), is increasingly recognized by AEC industry as an effective means to provide a mechanism of exchanging product-related data between different systems. It covers a wide variety of different product types and life-cycle stages (Pratt, 2001). Because of the complexity, the ISO-STEP standard had broken up into smaller parts that can be developed, balloted, and approved separately. Several hundred parts of the standard are issued. This makes STEP the biggest standard within ISO (Wang and Xu, 2015, Zhang et al., 2015b). These parts are referred to as ISO 10303-xxx, where xxx is the part number, and each part has its own scope and introduction in its own right. For example, parts (1x) are the description methods (EXPRESS, EXPRESS-X, etc.) while parts (2x) are the implementation methods (STEP-File, STEP-XML, SDAI, etc.) (Ridwan et al., 2012).

Industry Foundation Classes (IFC) and CimSteel Integration Standard-version 2 (CIS/2) are two main data models that have been developed based on the ISO-STEP technology and defined in the EXPRESS language. IFC is open and international standard “ISO 16739” (Zhang et al., 2015b) and a neutral data format for the whole buildings lifecycle to facilitate interoperability in the AEC industry. CIS/2 is a data exchange file format for structural steel design, analysis and fabrication supported by the American Institute of Steel Construction and the Construction Steel Institute in the United Kingdom (Lee et al., 2014).

The IFC is public and non-proprietary data model (Gupta et al., 2014, Mahdavi et al., 2014, Eastman et al., 2011b). It is the most comprehensive and widespread data exchange formats. IFC is therefore been found to be relevant in this research work.

3.3 Industry Foundation Classes (IFC)

IFC is an open data model and interoperable building information model precisely developed as a means to exchange model-based data among model-based applications (Steel et al., 2012b). It is the most comprehensive and widespread data model formally adopted worldwide by different governments and agencies as it can describe different building elements and multi-disciplinary designers (Gupta et al., 2014).

IFC is the most powerful standard available for tackling the existing challenges of interoperability (Zhang et al., 2015b, Gupta et al., 2014). From the viewpoint of model-based interoperability, IFC standard now is supported by most of the BIM applications (buildingSMART, 2016). Kiviniemi and Codinhoto (2014) emphasises that the robust of BIM software are linked to the developments of the IFC. This section includes an overview of the evolution of the IFC. IFC has been classified into two parts, a conceptual data schema, and an exchange file format for BIM data to clarify the descriptive and the implemental parts of IFC standard.

3.3.1 Overview of the IFC evolution

IFC is developed by buildingSMART (formerly the International Alliance for Interoperability, IAI). It is registered in 2013 with the ISO as an international standard “ISO 16739” (ISO, 2013). The development process of the IFC standard began in 1994 to create an interoperable format for representing buildings (Laakso and Kiviniemi, 2012). The first version was released in 1996. Since then, there have been ongoing releases and updates every few years to keep up with the growing demands of the industry. The latest version is IFC 2x4 (known as IFC4). It was released in March 2013. IFC4 specification contains 766 entities, 206 enumeration types, 1691 individual properties, 408 property sets, 128 defined types, and 52 select types (Gao et al., 2015). While these numbers indicate the expansive nature of IFC, they also reflect the semantic richness of IFC contents, considering multiple various applications (ranging from building information, structural analysis, energy analysis, cost estimation, scheduling, etc.…).

As mentioned earlier, the IFC specification is written using the EXPRESS data definition language based entity-relationship model, defined as ISO 10303-11 and the IFC exchange file structure is called STEP physical file format, defined as ISO 10303-21. The next two sections demonstrate in more details the EXPRESS data definition language and the STEP physical file format.

3.3.2 IFC EXPRESS schema

The specification of the IFC standard consists of the data schema, represented as an EXPRESS schema specification. EXPRESS is a data modelling language formalized in ISO standard for defining data objects and relationships among them. It is standardized in the ISO-STEP as ISO 10303-11. The EXPRESS language has been used in a wide range among different domain. The formal description language EXPRESS is not a programming language, but a specification language for the logical and consistent description of the information models of STEP in terms of entities, attributes, and constraints. For convenience, The EXPRESS schema specification of the IFC standard is simply referred to as IFC EXPRESS schema (or IFC schema).

IFC EXPRESS schema can be defined in two ways, textually and graphically. The textual representation is more important for the formal verification and as input for tools while the graphical representation, called EXPRESS-G, is more suitable for reader use (Schuler, 2001). The structure of a data model in EXPRESS-G can be presented in a more understandable manner compared with the textual representation, but not able to represent all details that can be formulated in the textual form (Arnold and Podehl, 1999).

3.3.2.1 Architecture of the IFC EXPRESS schema

IFC Express schema includes inheritance hierarchy of entities, which represent project information, project elements, features of the elements and the relationships among those entities. Each entity is within a complex sub-entity definition tree. The data schema architecture of the IFC is divided into four conceptual layers based on different concepts (such as elements, properties, designers, process, etc.…). Within each conceptual layer, a set of entities schema are defined (Mikael Laakso, 2012). Figure 3.2 shows the IFC schema architecture. The layering system is designed in such a way that an entity in any conceptual layer can only be related to or referenced an entity in the same or lower layer, but not an entity from a higher layer. The layering system is intended, to make the IFC model easier to maintain and grow, to allow lower-layer entities to be used again in higher-layer definitions, and to make a difference between multi AEC disciplinary entities, so that the model can be more easily implemented in individual discipline-specific applications. A brief description of the conceptual layers is given here, starting from the lowest to the highest:

- Resource Layer: it contains different sets of supporting data structures. Each set has different categories of entities to represent an individual business concept (such as, Representation Resource, Geometry Resource, Profile Resource, etc.…) . Thus, all the features of an element are represented in this layer. The resources form the lowest layer in IFC model architecture can be used or referenced by entities in the other layers. For example, the resource data schema (*IfcGeometryResource*) contains entity (*IfcCartesianPoint*) to define the coordinates of a point that will use it later to define the location of a building element.
- Core Layer: it contains the most abstract concepts within the AEC industry. This layer provides the basic structure, the fundamental relationships, and all further specializations in aspect specific models. Entities defined in this layer can be inherited by all entities above in the hierarchy. Entities in this layer can be divided into two levels of generalization (kernel and Core Extensions).

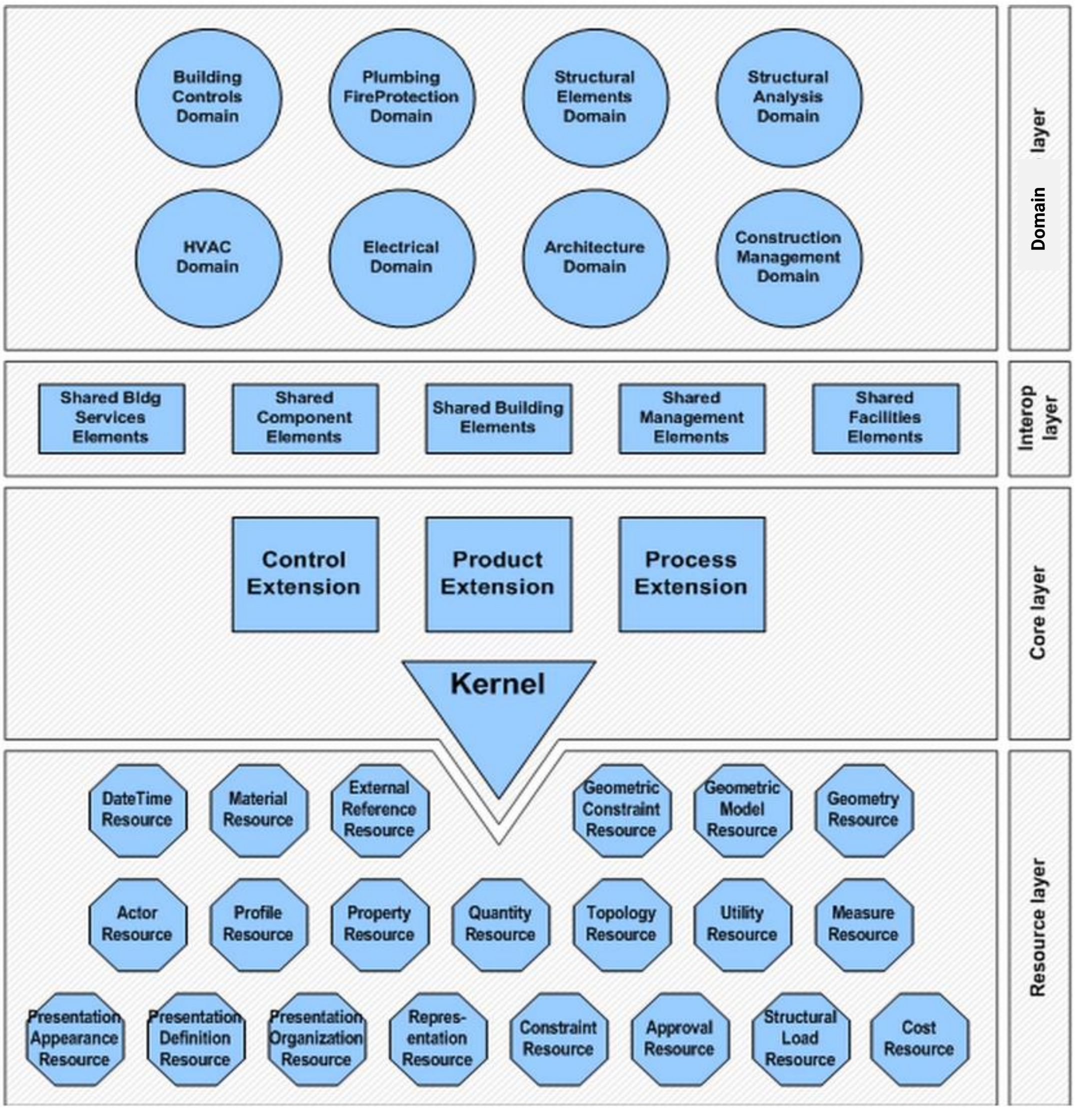


Figure 3.2: IFC schema architecture with conceptual layers (Building-SMART, 2015)

- The kernel level (*IfcKernel*) defines the most abstract part of the IFC architecture, like object, property, and relationship. *IfcRoot* entity is the top of the entity hierarchy of this part. It has information of identity (Global Unique Identifier "GUID"), together with attributes for name, description, and owner history. *IfcRoot* is divided into three abstract concepts: *IfcObjectDefinitions* (to capture tangible object occurrences and types, such as beam, foundation, task, person, work order), *IfcPropertyDefinition* (to capture some properties about objects), and *IfcRelationship* (to captures relationships between objects, object-property). These three entities are the base to define the second level of specialization when more detailed information regards the actors, controls, groups, elements; processes, etc.… are defined.

  - The Core Extensions level (*IfcProductExtension*, *IfcControlExtension*, *and IfcProcessExtension*) further specializes the IFC concepts. The Product Extension schema defines abstract components in the AEC industry such as space, site, building elements, annotation, etc. The other two Extension schemas define process and control related concepts such as task, procedure, work schedule, performance history, work approval, and so on (Deng and Chang, 2006). The Product Extension (*IfcProduct*) is the focus of this study because it is the base entity for all physical objects (building elements, structural analysis items, etc.).

- Interoperability "shared" Layer: This level comprises entity categories that are commonly used and shared between multiple building construction and facilities management applications. Thus, the shared building elements "*IfcSharedBldgElements*" schema has entity definitions for most of the common building entities, such as beam, column, wall, slab, door, etc. . . .
- Domain Layer: The highest level of the IFC model contains entity definitions useful in a specific domain. This layer organizes definitions according to industry discipline, such as architecture, structural engineering, facilities management, and so on. Thus, the unshared building elements can be represented in this layer. For examples, footing, pile, and reinforced bars are building elements that have structural nature. *IfcStructuralElementsDomain* entity

represents the schema of the structural elements and so on for the other domains. Entities defined in this layer are self-contained and cannot be referenced by any other layer.

IFC distributes all entities into rooted and non-rooted entities. Rooted entities are the entities that represent the building elements and their relationships. They derive from *IfcRoot* and cover the core, interoperability and domain layers. The *IfcRoot* entity is the top of the entity hierarchy and it is the parent of all rooted entities, all of which have globally unique identifiers. Non-rooted entities cover the resources layer and only presented if referenced from a rooted instance. Non-rooted entities may be used multiple times within the rooted tree. All building elements within the IFC EXPRESS definition (for instance, *IfcFooting*) serve as child entities in the inheritance hierarchy that should be followed back to the object definitions (*IfcObjectDefinition*) and then to the root entity "*IfcRoot*". Figure 3.3 shows an EXPRESS-G diagram of the entity "*IfcFooting*", its parent entities, inheritance attributes, and IFC architectural layers.

It can be observed from the figure that to define *IfcFooting*, a list of entities needs to be defined starting from the *IfcRoot* entity. Thus, the inheritance hierarchy of the footing entity *IfcFooting* derived from the building elements schema (*IfcBuildingElement*), which in turn is a sub-entity of the element entity (*IfcElement*) and so on going the way up to the root entity (*IfcRoot*). In addition, entities related to the resources layer are not with the rooted hierarchy, but can be used or referenced by entities in other layers. Moreover, the rooted entities (domain, interoperability, and core layers) can inherit based on the classification in the IFC model architecture from the highest to the lowest levels.

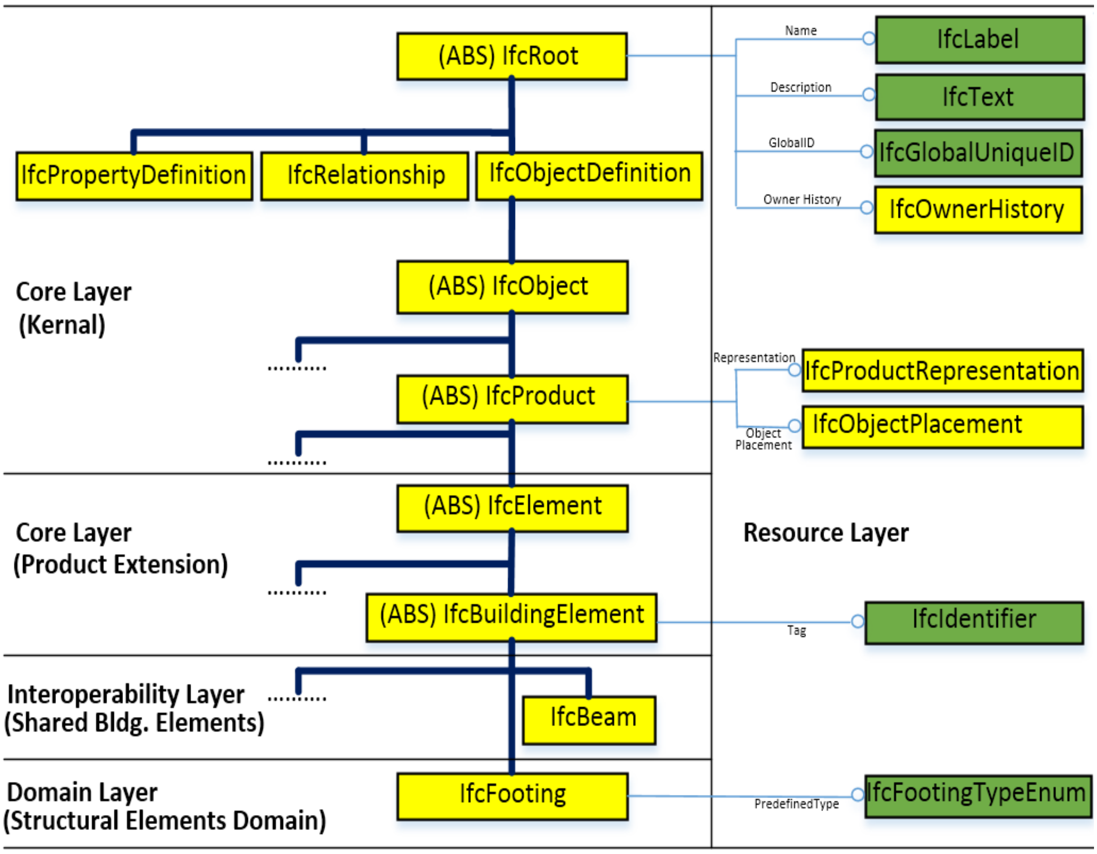


Figure 3.3: *IfcFooting* entity with its attributes using EXPRESS-G.

Attributes may be associated with each IFC entity. Each attribute is defined directly by a numeric or descriptive values, which in turn is defined by a particular collections including set (unordered), list (ordered), or array (ordered, sparse) or references to a particular entity. The entities inherit all the attributes from the parent entities. For example, (Figure 3.3), *IfcProduct* has representation and object placement attributes, both are inherited by the footing entity (*IfcFooting*), which is a child entity to the super entity *IfcProduct*. Therefore, all sub entities hierarchy of *IfcProduct* inherit the representation and object placement information within the project structure. These two attributes provide the beginning of defining the geometric description of any building element entity in the IFC schema.

3.3.2.2 The Relationships within IFC EXPRESS schema

In addition to the object definitions (*IfcObjectDefinition*) and feature definitions (*IfcPropertyDefinition*) that respectively define different elements and features (Figure 3.3), the relationships between elements and/or features, which are defined as *IfcRelationship* entities, can be identified in IFC. There are rich set of relations represented within the subtypes of *IfcRelationship*, below are the classification of the *IfcRelationship* with an example to each of them:

- *IfcRelAssigns* defines assignment between object entities (assigning an object "product" to a set of other objects that are subtypes of *IfcObject*).
- *IfcRelConnects* defines connections between object entities under some criteria (connecting the geometric shape representation of two elements).
- *IfcRelDecomposes* defines composition relationships between object entities (aggregation of beams and columns in the structural frame).
- *IfcRelAssociates*, associations between objects to sources of information (approval, classification, etc...) (associating the material information to a *set of elements* (subtype of *IfcObject*))
- *IfcRelDefines*, relationships from objects to objects describing the property or type information (different beams sharing the same properties).
- *IfcRelDeclares*, declaration of objects or properties to a project (*IfcBeam* as a building element is declared within the context of *IfcProject*).

3.3.2.3 Project Information in IFC EXPRESS schema

Identify the information related to the project of the building model from the IFC schema is necessary to map the changing information in this research work.

3.3.2.3.1 Ownership Information

The *IfcOwnerHistory* entity captures the information about the owner who deals with the current model, the organization that generate the file, the software application that has been used, the creation date and time as well. It is a direct attribute to the root entity (*IfcRoot*) (Figure 3.4). Therefore, it is indirectly attached to all objects, relationships, and properties entities. Many entities are derived from *IfcOwnerHistory* that can be used in inheritance and reference relations. Table 3.1 presents the general information required within *IfcOwnerHistory* about the current data model. The list of predefine all actors or human agents involved in a project during its full life cycle are enumerated in *IfcActorRole,* table 3.2 illustrate the existing disciplines in the IFC model:

Table 3.1: General information within *IfcOwnerHistory*.

Entity	**Description**
IfcApplication	Holds the information about an IFC compliant application (Application Developer, Version, Application Full Name, and Application Identifier).
IfcChange ActionEnum	identifies the type of change that might have occurred in the exchanged model
IfcStateEnum	Identifies the state or accessibility of the object (read, write, locked).
IfcTimeStamp	The date and time measured as the number of seconds elapsed since 1 January 1970.
ThePerson	User who carried out the last modification
IfcOrganization	organization who carried out the last modification
IfcActorRole	a role which is performed by an actor

Table 3.2: Existing disciplines within *IfcActorRole*

ARCHITECT	MECHANICAL ENGINEER	CONTRACTOR
CIVIL ENGINEER	ELECTRICAL ENGINEER	CLIENT
MANUFACTURER	BUILDING OPERATOR	OWNER
SUBCONTRACTOR	FACILITIES MANAGER	SUPPLIER
PROJECT MANAGER	STRUCTURAL ENGINEER	ENGINEER
COST ENGINEER	COMMISSIONING ENGINEER	RESELLER
USER DEFINED	CONSTRUCTION MANAGER	CONSULTANT

The IFC EXPRESS has a mechanism that supports *IfcOwnerHistory* entity for capturing the most recent changes for a specific object. An enumeration type *IfcChangeActionEnum* within the IFC schema defines the actions associated with the recent changes made to the objects (such as, 'added', 'modified'). *IfcOwnerHistory* is the only entity that references *IfcChangeActionEnum*. This means that it is used with the entities stemming from *IfcRoot* (such as, beams, columns…), but not with the entities that belongs to the resources data schemas. An investigation to the exchange of *IfcOwnerHistory* between software applications was done by (Liebich et al., 2008) using IFC compatible applications. He found that *IfcOwnerHistory* entity in the IFC STEP exchange format of most tested software applications (ArchiCAD, ADT, Revit, Allplan and MagiCAD, except Solibri Model Checker) does not deal with change action mechanism and does not exceed being a dummy object. In another study, van Berlo and Krijnen (2014) mentioned that from 122 different IFC models, only 2 contain multiple owner histories. Even within these two files, it does not associate information on responsible users to the products in the model.

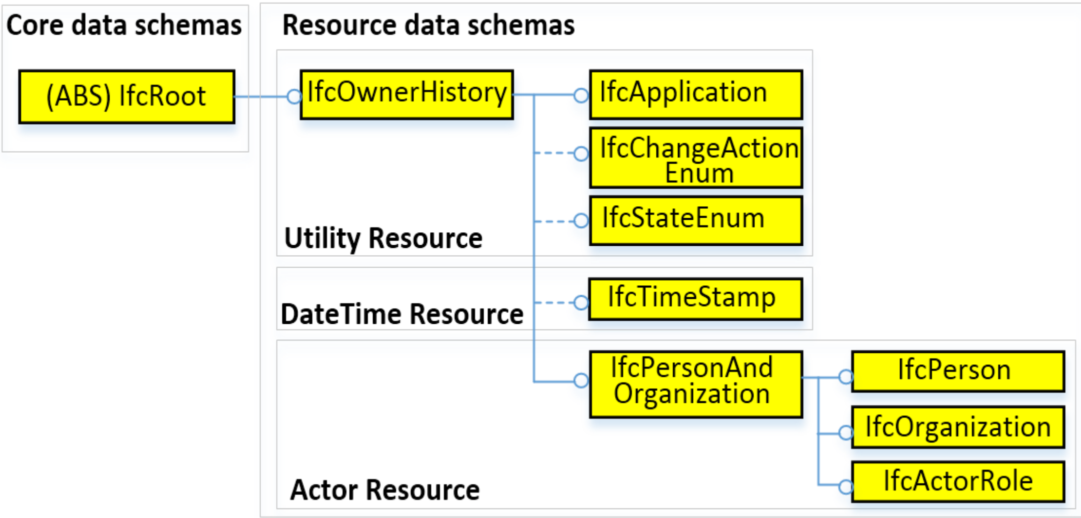


Figure 3.4: EXPRESS-G for ownership entity, *IfcOwnerHistory*.

Beside the non-use of all the characteristics of the *IfcOwnerHistory* effectively in most of the software applications, there are two main limitations of this entity. The first limitation is that the entities belong to the resources data schemas, which are responsible to represent the element features (shape, material, etc...), cannot link with the *IfcOwnerHistory* entity. Thereby, the feature-level for the element does not capture information regarding the type of the change that might have occurred during the last session. Only the building element-level that are inherited from the *IfcRoot* can hold the change information. So that, the profile section of the beam, for example, cannot be considered as change information in the IFC standard.

The second limitation is that the *IfcChangeActionEnum,* which is referenced only by the *IfcOwnerHistory,* defines the last change type that might have occurred to the object but not the change value (text or number). As a result, the current IFC standard does not capture the whole design changes. Only an indication of the change at a very basic level in the building can be clarified.

3.3.2.3.2 Element Features

The element features (e.g., I-shape, coordinates, steel, area, cost, etc.…) that discussed in chapter 2 are needed to be investigated, analysed and clarified in the IFC schema to acquire the relation between the features and the relevant element. These features represent different sets of information (shape, location, material, quantities, specification, etc.) and not collect in one place within the IFC schema to be directly connected to the relevant element. Based on the data schema architecture of the IFC EXPRESS mentioned before, all shared building elements (*IfcBeam, IfcColumn, IfcWall, IfcSlab*, etc.) are derived from the *IfcBuildingElement* entity, which are within the rooted entities. Whereas, all element features are represented in the resources layer, which are within the non-rooted entities. The clarification of the link between the building elements with their features within the IFC schema is further discussed.

(i) Geometry shapes and spatial locations

Geometry shapes and spatial locations are permanent features within each building element and are one of the main features used to generate the building elements.

All building elements (e.g. beam, column, and slab) can have these two features indirectly under the main generation of the elements (e.g. *IfcBeam*, *IfcColumn*, and *IfcSlab*). This information can be derived from the two attributes (*IfcProductRepresentation and IfcObjectPlacement)* located within the inherited entity *IfcProduct*. The first attribute is to define the geometric property of the product and the second attribute is to define the placement in the spatial context. These two attributes are related to each other, all of the geometric representations contexts of the same object are defined within the same object coordinate system. The structure of extracting these two attributes is shown in EXPRESS-G in Figure 3.5. For simplicity, a beam element is used for analysing information.

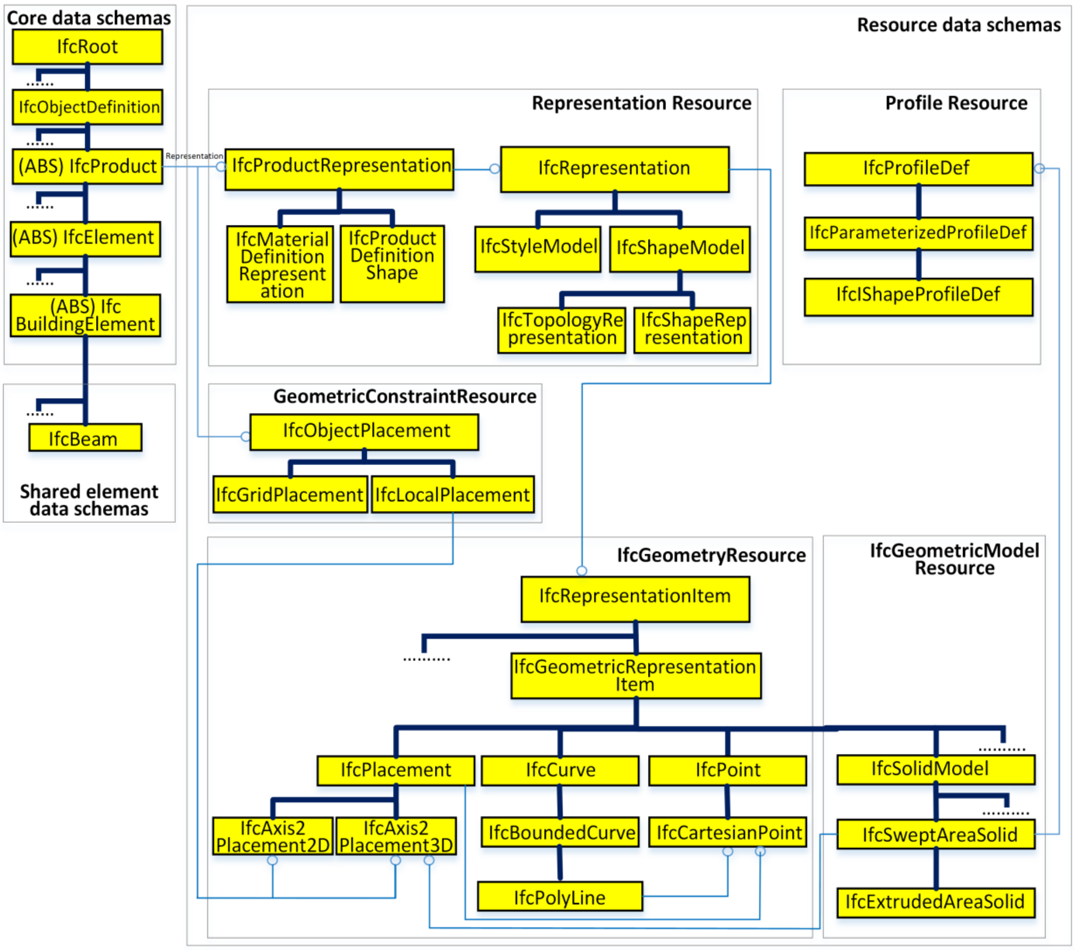


Figure 3.5: EXPRESS-G for geometry shapes and spatial locations

The sequence to identify the geometry shapes starts with the representation and characterization of the required information from the Representation Resource Layer. Figure 3.6 illustrates the sequence of steps required to represent and identify the geometry shapes. There are two sets of schemas that have to be supported for geometric representations of a product within the representation resource (*IfcProductRepresentation* and *IfcRepresentation*).

IfcProductRepresentation (and sub entity *IfcProductDefinitionShape*) is a general container for all representations for a product. It allows for a characterization of the product representation by a name and for a provision of further description information. *IfcRepresentation* (and sub entity *IfcShapeRepresentation*) is more specific container for product's representation. It carries additional classifications provided by Representation Identifier to denote the kind of the representation (e.g. Axis, Body, etc.) and Representation Type to define the geometric type used (e.g. solid model, surface model).

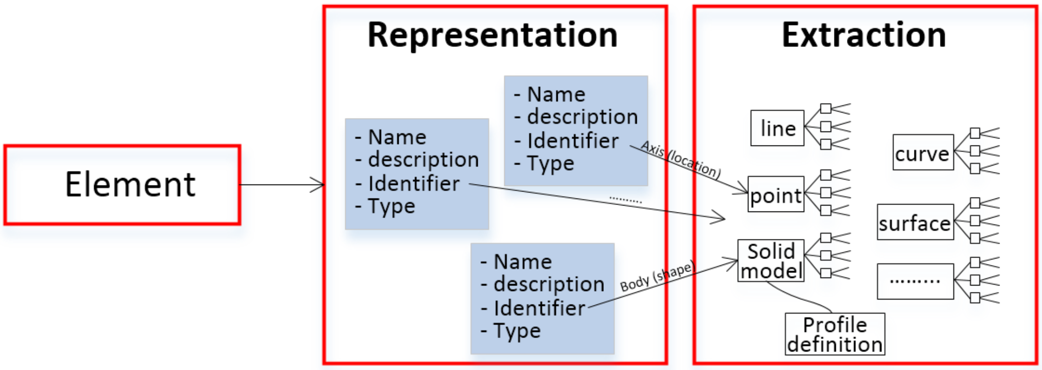


Figure 3.6: Sequence steps to represent the geometry shapes

After representing the geometric shape of a product, the geometry items can be allocated in the Geometry Resource Layer. *IfcGeometricRepresentationItem* is a super entity of all geometric shapes (e.g. point, curves, surface, solid, 3D solid object, etc.…). Since most of the used geometries for building elements are solid models, *IfcSolidModel*, as a sub entity of *IfcGeometricRepresentationItem,* is used to illustrate different 3D solid shapes of physical and spatial elements. *IfcSolidModel* is the top entity within the Geometric Model Resource Layer to define 3D solid shapes in different ways, such as Boundary representation "Brep", CSG representation "CSG", Sweeping representation "Swept Area Solid" and other solid representation (Table 3.3). For example, the "Swept Area

Solid" type requires a 2D planar cross section to sweep through space. The 2D cross section can be providing by profile definitions in the profile resource layer. *IfcProfileDef* is the top entity of all definitions of the section profiles in commonly used standards. For example, to select I-shape profile, *IfcIShapeProfileDef*, as a sub entity extracted from the parent entity *IfcProfileDef*, defines the parameters of all 'I' section profile.

Table 3.3: Shape representations for solid models

Solid shape Type	**IFC Entity (Derived from *IfcSolidModel*)**	**Description**
Swept Solid	*IfcSweptArea Solid*	Sweeping representation allowing 2D planar cross section to sweep through space.
B-rep	*IfcManifold SolidBrep*	Manifold solid boundary representation used Boolean operations (union, intersection, and subtraction).
CSG	*IfcCsgSolid*	Constructive solid geometry model. Represented by a single 3D CSG primitive, or by a tree of operations and algebraic expressions.
Advance dSwept Solid	*IfcSweptDisk Solid*	Sweeping representation allowing 2D circularly bounded plane to sweep through space.

In the representation of spatial locations, it can either be absolute placement (relative to the world coordinate system), relative placement (relative to the object placement of another product), or constrained placement (relative to grid axes). The default way to represent the location of a product is by using the relative placement, given by *IfcLocalPlacement*, as established at the Geometric Constrain Resource (Figure 3.5).

The next step is to define the location of an item (an entire shape). As mentioned before, the geometries for building elements are solid models.

Then, four entities (*IfcPoint*, *IfcPolyline*, *IfcSolidModel,* and *IfcPlacement*) that are extracted from the *IfcGeometricRepresentationItem* are the main entities to define the Cartesian points of the 3D solid shape. *IfcPoint* and its sub entity, *IfcCartesianPoint,* are responsible to define a single 3D coordinates. *IfcPolyline* is a bounded curve that represents the object as linear segments defined by a list of Cartesian points. If there are two Cartesian points to define the object, then the polyline is closed curve (Karstila et al., 2001). *IfcSolidModel,* as explained before, is used to define different shapes for solid model of the elements. *IfcPlacement* locates a geometric shape with respect to the coordinate system of its geometric context (Zhao, 2012). The four entities listed above are all linked together to define the 2D and 3D Cartesian points of a 3D object.

(ii) Material Information

A homogeneous or inhomogeneous substance can be used to form building elements in IFC model. *IfcMaterialDefinition* is the top entity within the Material Resource schema. It can define and collect different types of material characterization. Associate the material definitions to the related building element in the IFC model can be classified into three steps (Figure 3.7).

- Define the materials within the Material Resource schema. There are different ways to define the material in IFC: by a single solid material (such as a typical beam), by a number of layers (wall composed of brick, foam ‘insulation’ and wood), by varying profile shape (column with different cross sections) or by a number of parts of an element, each part has an individual material (doors with components such as lining, framing and glazing). Table 3.4 shows the IFC entities for each element type. *IfcMaterial* is the basic entity for material designation and definition; it is used with the other three types above to form one solid material.

Table 3.4: IFC entities for each material type

Material type	**IFC entity**
single solid material	*IfcMaterial*
number of layers	*IfcMaterialLayer*
varying profile shape	*IfcMaterialProfile*
number of parts of an element	*IfcMaterialConstituent*

- Collect the defined materials in a group (if there is more than one material to form the element), this is done within the *IfcMaterialDefinition* entity.

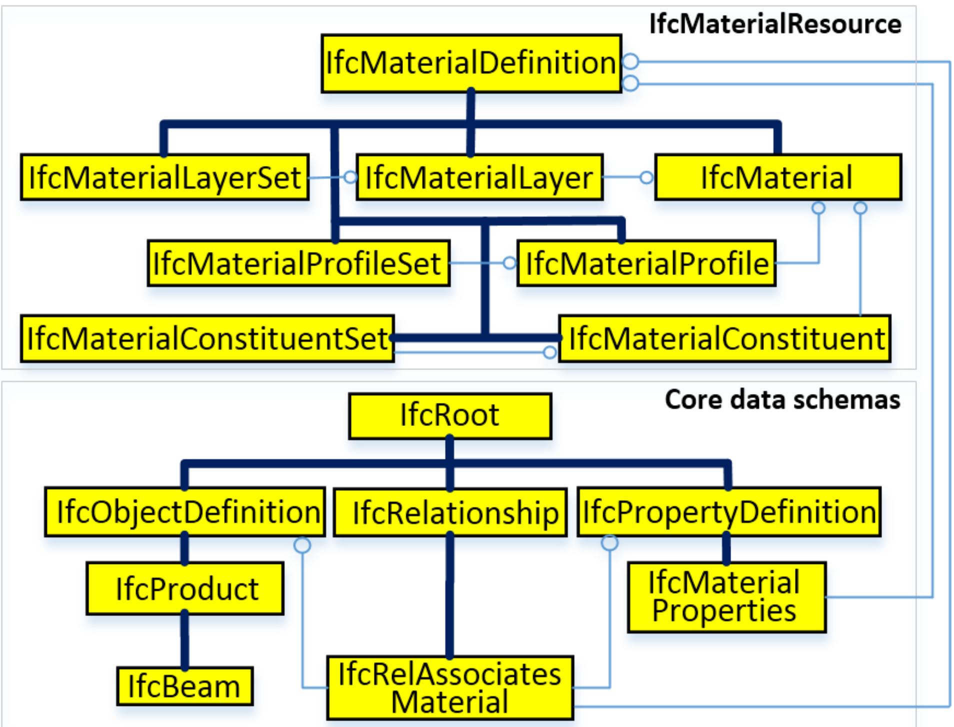


Figure 3.7: EXPRESS-G for material information of a product

- Assign the collected materials to the related building element by using a relationship entity within the *IfcRoot* schema. An objectified relationship entity *IfcRelAssociatesMaterial* is used to associate the material definition and the elements to which this material definition applies. This relationship entity allows the material(s) definition to be assigned to one or many building elements.

(iii) Additional features

In addition to the geometry shapes, spatial locations, and material information, each building element can also hold other types of features that users might want to exchange (such as information that support quantities, cost estimating, thermal transmittance, construction planning, facilities management, etc.…). There are no specific entities within the IFC model to define these features. Since there are numerous alphanumeric attribute definitions depending on life-cycle stage, discipline, building regulation and region, there will never be a complete agreed internationally standardized attributes (Zhiliang et al., 2011).

For this purpose, the IFC model defined a flexible and powerful mechanism to allow extending the IFC model through the feature Definition mechanism. It can be either existing definitions that are shared among single or multiple elements, or extended definitions to a library that are added by the end users. Assigning the feature definitions to the related objects in the IFC model can be classified into three steps (Figure 3.8):

- The first step is to define the feature types and their values. *IfcPropertyAbstraction* is an abstract super entity of the existing and the extended features in the Property Resource schema and *IfcProperty* is a common entity for all features types within the IFC standard. Table 3.5 shows all feature types that can be associated with the IFC. For example, a sub entity *IfcPropertySingleValue* is a general entity to define a single feature object as a (feature - single value) combination that provide a feature name, a description, and a nominal value. This entity is the most used concept for exchanging property.

- The second step is to collect the defined features in the first step as groups. All defined feature entities within the Property Resource schema gather by a container entity (*IfcPropertySet*) to facilitate more flexible and easier association between IFC objects and a set of features (P-set). Wix et al. (2008) defined feature set as a collection of free attributes. There are collected feature sets for many types of building elements, such as beam, column, window glazing, and reinforcement.

Table 3.5: Property types within *IfcProperty*

Feature type	**IFC Entity (Derived from *IfcProperty*)**	**Description**
Property with Single Value	*IfcProperty SingleValue*	Property that has single value assigned of the same type.
Property with Enumerated Value	*IfcProperty EnumeratedValue*	Multiple property from a predefined list of selections
Property with Bounded Value	*IfcProperty BoundedValue*	vary value between an upper and lower limit
Property with List Value	*IfcProperty ListValue*	Property that has several values assigned of the same type.
Property with Table Value	*IfcProperty TableValue*	Set of values .Each value stored is dependent on another value.

- The third step is to link the feature collections to the relevant element. All defined feature sets are linked to the objects using the relationship entity *IfcRelDefinesByProperties*. The relationship mechanism allows for the product and the feature definitions to exist independently and link the *IfcPropertyDefinition* to the *IfcObjectDefinition*.

By using this approach, it is possible to define many feature information, gather them by different feature sets, and link these sets to the relevant

IfcProduct entity. Then the connection between the building elements with many features (not the shapes, locations, and material) within the IFC schema can be identified.

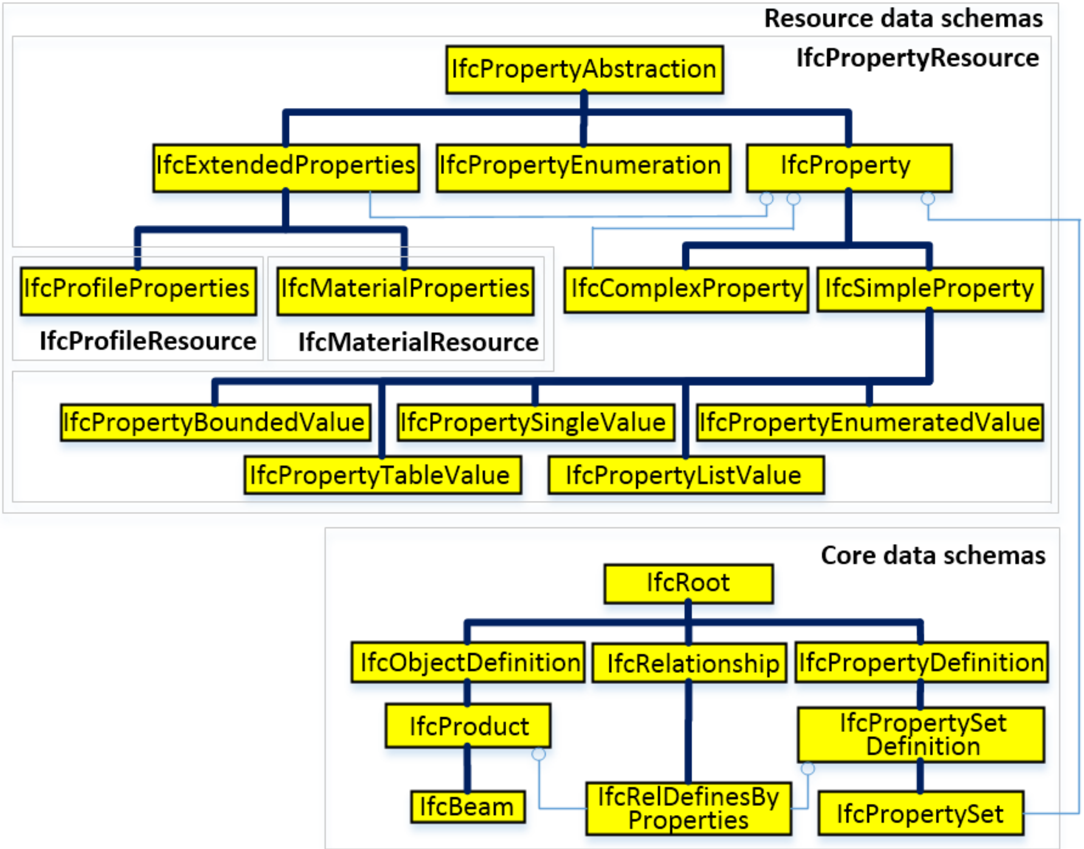


Figure 3.8: EXPRESS-G for additional features of a product

3.3.3 IFC STEP-File

ISO-STEP is primarily defining data models using the EXPRESS modelling language. Application data defined according to the IFC data model can be exchanged using multiple IFC file formats, including text, XML, and zip. All the formats are based on the ISO-STEP standards.

- STEP-File “IFC-SPF” is a text format defined by ISO 10303-21; it is an ASCII file format where each line typically consists of a single object record.

- STEP-XML "IFC-XML" is an XML format defined by ISO 10303-28. This format is suitable for interoperability with XML tools and exchanging partial building models.
- STEP-ZIP is a ZIP compressed format consisting of an embedded STEP-File and STEP-XML.

Due to the large size of typical building models, a standard exchange file format "STEP-File" is more common in practice than the other IFC file formats (Chao and Kim, 2015, Sun et al., 2015). This format is adopted in this work to exchange IFC between different applications. For convenience, the STEP-File format is referred to as IFC STEP file (or IFC file).

The IFC file ".ifc" defines the encoding mechanism on how to represent data according to a given conceptual schema "EXPRESS" by using clear text encoding of the exchange structure (Sun et al., 2015). In other words, the IFC file is the implementation method to the IFC schema, which is the description method. Based on ISO-10303-21; the IFC file splits into two sections (a header and a data section). The header section has some general information concerning the project, e.g. the IFC release, filename, author, date, and organization. The header section has a fixed structure consisting of 3 to 6 lines in a given order, which is very short relative to the other section. The second and main section of the IFC file is the "data section". This section contains all instances for the entities of the IFC specification that represent the current engineering project (Sun et al., 2015). Due to its ASCII structure, it is easy to read with typically one instance per line. Each instance line (called entity instance) in the exchange structure represents one specific EXPRESS schema. It has a unique STEP number in the form "#1234" (called instance number) to define the instance and to reference the other entity instances through the explicit attributes (Lipman and Lubell, 2015).

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION((' View Definition [Coordination View]'),'2;1');
FILE_NAME('simplehouse2.ifc','2015-04-13T15:42:19',(''),(''),'Autodesk Revit 2013');
FILE_SCHEMA(('IFC2X4'));
ENDSEC;

DATA;
#1= IFCORGANIZATION($,'Autodesk Revit 2013',$,$,$);
#2= IFCAPPLICATION(#1,'2013','Autodesk Revit 2013','Revit');
#3= IFCCARTESIANPOINT((0.,0.,0.));
#5= IFCDIRECTION((1.,0.,0.));
.
.
#652= IFCSHAPEREPRESENTATION(#43,'Axis','Curve2D',(#650));
#654= IFCPRODUCTDEFINITIONSHAPE($,$,(#652,#644));
#656= IFCBEAM('1J4cDH8r91mRE8mbTHxvqg',#52,'UB:UB305x127x37'171562');
#658= IFCPROPERTYSINGLEVALUE('Work Plane',$,IFCLABEL('Level : Level 2'),$);
#660= IFCPROPERTYSINGLEVALUE('Start Extension',$,IFCLENGTHMEASURE(-12.7),$);
IFCPROPERTYSET('2a$hanVLzBi9HegS0HDphq',#52,'Constraints',$,(#658,#660,#439,#));
.
.
#82393= IFCRELCONNECTSPATHELEMENTS('04KQJRWh5CDOqlq4L7tWb_',#52,END.);
#82395= IFCPRESENTATIONLAYERASSIGNMENT('A210',$,(#776,#792,#874),$);
ENDSEC;

END-ISO-10303-21;
```

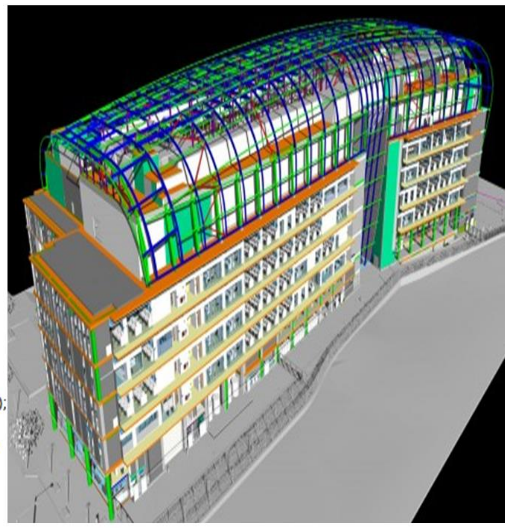


Figure 3.9: Excerpt IFC STEP physical file format and IFC BIM model (TEKLA-Company, 2015)

There are wide variety of BIM software that supports the IFC file format. This has mostly been in the form of importing from or exporting to IFC file. The buildingSMART (2016) website provides a list of 183 IFC compliant commercial software applications available for a variety of AEC applications (architectural, structural, services, building performance, construction management, data server, geographic information system, model viewer, others). Beside the commercial software packages there is also a number of free software tools that support IFC (IfcWiki, 2015). As an illustration for how IFC data can be interfaced with in practice, Figure 3.9 depicts excerpt from IFC STEP file for information containing about a commercial building and visualizes what the same complete file looks like instantiated in BIM software. The size of a STEP file is usually quite large. For example, a simple model (four columns and four beams) would contain 780 lines of data in the STEP file when exported using Autodesk's Revit software package (Jaly-Zada and Tizani, 2013) .

3.4 Related Researches based on IFC

Due to the deficiency of the IFC relating to the challenge in covering various domains in the AEC industry, researchers and vendors have been urged to improve the quality of IFC. They have been scrutinizing the IFC standard to obtain a great benefit from its concept in order to expand the BIM scope.

Three areas of research based on IFC are suggested. The first research area is related to extracting information from the IFC file and then synthesizing it in different domains, this area of research is concerned more with dealing with the information in the IFC STEP file. The second area is related to extracting and adding information respectively from and to the IFC STEP file; this concerns extending the structure of the IFC EXPRESS schema. The last area of the research generally covers issues related to managing and sharing the IFC file. Below are further discussed these research areas:

3.4.1 Research on IFC STEP file

Research in this area included extracting information from the IFC STEP file and using it in other areas. Ma et al. (2013) suggested semi-automatic and specification-compliant cost estimation based on the IFC file of the design model. A prototype software application "BIM-Estimate" was proposed to estimate the cost of building project for tendering in China. The application is limited to a fixed number of building elements and following a specific Chinese standard for cost estimating. Oti and Tizani (2014) utilized an IFC file to capture information for analysing sustainability related information to inform decisions at the early stages of the structural design process. Gupta et al. (2014) proposed a standardised process of using a conceptual multi-model framework involving the IFC file to extract the data requirements of solar PV simulation models. Lee et al. (2011) investigated the compatibility and differences between the IFC file of the same components produced using different BIM applications such as ArchiCAD and Revit Architecture. The outcome of the investigation on a simple building shows that the two applications have almost 78% of the same entities. Wang et al. (2014) studied the stability of information exchange between architectural and structural disciplines through IFC file -based software to form structural model. Nour (2009a) developed a graphical user interface (GUI) to enable users to define exchanged data for partial model and export it in IFC-STEP file.

3.4.2 Research on IFC EXPRESS Schema

The availability of the current IFC schema does not provide a sufficient condition for interoperability. It can support a limited number of domains and scopes in the AEC industry and more developments are required (Ma et al., 2013). Weise et al. (2009) suggested two mechanisms to extend the IFC EXPRESS: new entities or types definitions, or using the property sets. Ma et al. (2013) illustrated that defining new entities or types is the best way to extend the IFC standard since the newly defined entities and types were be within the schema of the IFC EXPRESS.

This section considers the various approaches that may be adopted for the extension of the IFC EXPRESS standard. The extension is based on analysis of the gap that exists between concepts that need to be incorporated for the extension model development and concepts that already form part of the IFC EXPRESS. There are two scenarios of approach adapted by the researchers in this area:

- Extension of existing concepts
- Adding new concept

3.4.2.1 Extension of Existing Concepts

Some concepts in the IFC EXPRESS schemas already exist but they need extension to capture additional information about the concept. The extended information might be new or a modification to the current information in the existing concept.

The definitions of the IFC specification are not enough to represent the concept of structural member in terms of bridge, road, and tunnel engineering. Lee and Kim (2011) suggested a set of entities to modify or add to the IFC resources to represent the spatial and physical components of the above structure types. Ji et al. (2011) presented a new geometric representation of bridge structures. The new schema is integrated into the current IFC-Bridge schema to solve the data interoperability problem. The extended IFC-Bridge schema is evaluated in the applications between bridge design and structural analysis systems. Cemesova (2013) suggested new property sets to be defined for existing entities '*IfcSystem*' and '*IfcEnergyConversionDevice*' to support the design of low energy buildings.

To manage the cost data on the basis of material analysis information, an extension is proposed by Gökçe et al. (2012) in the IFC EXPRESS to cover the construction material cost information. A set of new entities with new relations in the frame of the Construction Resource Concept are proposed. Kléos et al. (2012) defined new attributes to the existing entity "*ifcStructuralLoadGroup*", this entity is responsible for defining load groups, load cases, and different combinations. The new attributes allow the load safety factors used by many codes of practices to be defined to hold the maximum and the minimum design load from different load cases.

3.4.2.2 New concepts

New concepts mean that the extension model development specifies information requirements that are not captured in the schema of the IFC EXPRESS. Therefore, suggested entities or attributes for new concepts need to be fully defined, including the connection to the other parts of the IFC EXPRESS.

The extension of the IFC by adding different domain models is the most interesting among researches. A methodology to add a domain to the IFC has been suggested by (Liebich and Wix, 1999). It contains describing a set of assertions linked to process models, domain requirements, and task descriptions to define a methodology applicable for industry-wide and commercial use. Weise et al. (2000) was one of the first in this area. The main requirements of structural engineers were captured and suggested to be integrated in the IFC framework, which were not supported in the IFC standard at the time. The proposed extensions included adding new building elements *IfcDeepFoundation* and *IfcShallowFoundation* to represent the foundation elements. Moreover, he defined some structural analysis concepts, such as *IfcStructuralAnalysisModel*, *IfcStructuralConnection* and *IfcStructuralRepresentation*. Most of the proposed entities have been formally accepted in IFC2x3 release.

For the purpose of allowing the IFC file to represent as-built and as-damaged information, Akinci and Boukamp (2003) proposed to merge design and as-built information in one IFC file. A new entity *IfcRepresentationContext* has been suggested to store different representation contexts for a building element, ‘Design’, or ‘As-built’. This entity has been used for each product representations that have design and as-built information. Ma et al. (2015) proposed an as-damaged data model based on the IFC schema to represent the damage modes of RC structures. He suggested representing the damaged element into two or more segments. To achieve this, two new entities have been suggested: *IfcBuildingElementSegment* models parts of damaged elements; and *IfcRelSegments*, models the objectified relationships between original building element and their segments.

3.4.3 Research on Different IFC Domains

IFC aims at supporting data exchange and sharing among the various participants in a building construction or facility management projects. Many researches dealing with different domains (such as central shared model, change management, and versioning) have shown how IFC can support the AEC environment.

Redmond et al. (2012) emphasized that the main feature of BIM is its ability to share synchronize information across multiple software applications through using IFC. The IFC specification is the best attempt made to provide support for the idea of collecting all information of a building model in a shared representation. Figure 3.10 illustrates two options to how a number of different software applications translate information directly or share information using the neutral IFC data model. Other researches (Gielingh, 2008, Laakso and Kiviniemi, 2012) confirmed that the scenario of everyone communicating with everyone directly is excessive and not representative of the actual data exchange needs for the AEC industry. Chen et al. (2005a) provided a simple case study to show how an architect could collaborate with a structural engineer through the use of the IFC-based web server and how information flows at the collaborative design phase. Plume and Mitchell (2007) demonstrated that the IFC file can be loaded into a STEP model server to hold the building model as an object database on a central shared computer and accessible across the Internet.

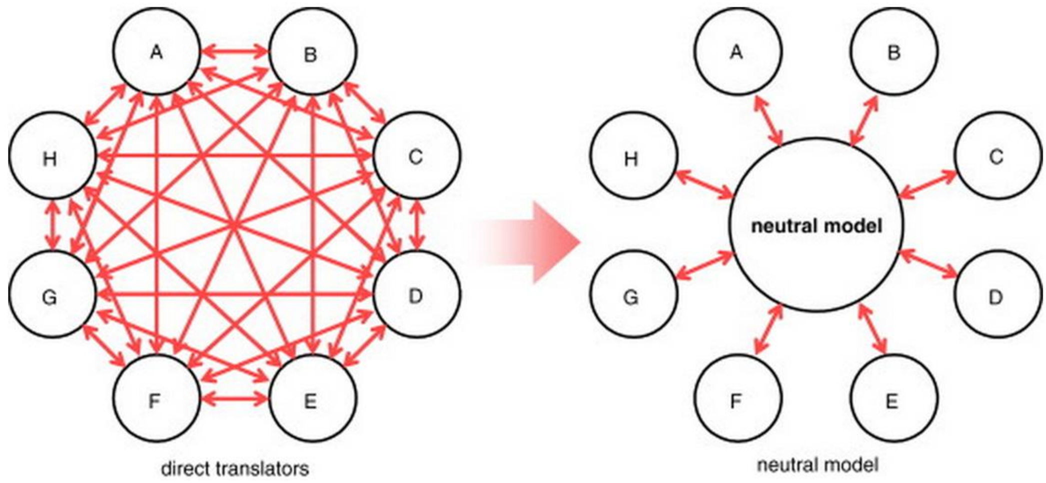


Figure 3.10: Direct translators vs. neutral IFC data model (Gielingh, 2008).

Few researches have covered the area of managing the design changes; most have covered the changes in the contents of the IFC EXPRESS releases. Amor and Ge (2002) presented a process of generating mappings between different IFC EXPRESS releases to map a population of the schema from one release to another. Wang et al. (2007) presented a semi-automated approach for detecting the differences between two IFC models. This approach incorporates taxonomy for describing differences between the two releases. Two test cases had been discussed include identification of differences between: IFC R1.5.1 and IFC R2.0; and IFC R2x and IFC R2x2. Nour et al. (2006) proposed a database for classification the objects of the IFC model using the identifiers of the instance line that represent the entity and IFC GUID. Amor and Ge (2002) developed a system to interrogate two schema versions in the same domain and generate a mapping specification between them based upon recognizing a classification of the relationships between entities and types in the relative schemas. Nour and Beucke (2008) addressed the problem of binding the growing number of IFC versions and their EXPRESS definitions to programming environments. They developed an automated process of generating early binding classes for a new version of the IFC model. Nour and Beucke (2010) clarified that versioning the objects of IFC file can help overcome many problems in change management system. Experimental tools were developed in this work to handle the BIM updated based on the GUIDs of the IFC objects. The limitation in this work is the GUIDs are not provided in the instances representation of the resource layer.

3.5 BIM applications and Design Changes

IFC is a BIM that is open and interoperable between varieties of BIM applications. A number of researchers, practitioners, software vendors, and professional organizations are working hard to develop suitable solutions to deal with the challenges of the design changes. Few of BIM applications include some privileges to deal with the design changes. The applications that are dealing with changes can be classified as follows:

3.5.1 Using BIM models:

Some stand-alone BIM applications have built-in the advantage to manage the design changes through their applications. The BIM model has to be generated by the same BIM application only. For instance, Autodesk Revit products provide tools to monitor the linked models and coordinate changes among the teams that are using the same models type (.rvt) (Pilehchian et al., 2015). When one team changes a monitored element, other teams are notified by a warning message so that they can adapt their designs or work with team members to resolve issues (Revit, 2015). The drawbacks of this method are that not all the elements can establish relationships between them (only columns, floors, and walls in addition to the grids and levels) and the warning message displays every time in each linked models. Graphisoft ArchiCAD products allow of compare different disciplinary models of the product (.pln), and detect and highlight the differences of the building elements between the two selected versions of ArchiCAD models.

3.5.2 Using IFC models:

The models generated by different BIM applications are differ in their structure and capabilities to establish an interoperable models. IFC serves as an intelligent and universal data model to exchange incompatible files of the BIM applications. Many of the BIM applications, as an export and import option, already use the IFC data model (e.g. Revit, ArchiCAD, etc.…). Solibri Model Checker (SMC), Navisworks, Tekla BIMsight, ArchiCAD, etc. applications are information modelling integrated with IFC data modelling to mainly manage BIM models, detect clashes and resolve conflicts among different objects for two disciplinary models (Volk et al., 2014). The both model versions of the project must be saved as IFC files and the comparison is based on the GlobalId numbers in each of the two versions. For instance, Solibri Model Checker (SMC) uses IFC as the basis for collaboration among disciplinary designers to manage changes between two design models through compares design models and highlights the clashing components (SOLIBRI, 2014, Solihin and Eastman, 2015). These applications are commercial standalone and proprietary products mainly for tracking current changes between the older and newer IFC models with easy visualization of model changes.

Construction projects usually involve different disciplinary teams. Many issues arise during the design process by one of the teams, which need to be exchanged to one or more of the other teams. That's where IFC comes in; exchanging the actual models via an ‘open standard’. But the IFC is only storing information, and rising and transmitting the issues found is not supported with the IFC standard (for instance, reporting the detected design changes). Therefore, IFC model is not suitable for documenting issues or comments, and providing clear communication (Linhard and Steinmann, 2014). Tekla Corporation and Solibri Model Checker (SMC) in 2009 have developed BCF (Building Collaboration Format) as an open file format that allows the addition of textual comments, screenshots and more on top of the IFC model layer for better communication between BIM software tools (Zhang et al., 2015a). Now, the BCF has been submitted to ISG (Implementer Support Group) of Building SMART under the new Affiliation Scheme to become an official buildingSMART specification. It is based on XML and can be implemented as a web service (van Berlo and Krijnen, 2014). BCF is supported by some BIM tools (such as, Solibri Model Checker, MagiCAD, Tekla Structures, Tekla BIMsight, DDS, and some other)(Shafiq et al., 2013) .

The BCF file format does not identify or track the changes in the BIM models. It helps designers to keep track of design issues as design evolves and gets fine-tuned, and it saves time by sending a clear message about changes across disciplines. BCF idea is to select the required elements manually and encode messages containing (raise issues, attach snapshots, provide answers, propose suggestions, and change requests) through using IFC mechanisms for Global Unique ID's (GUIDs) (BuildingSMART, 2015).

The BCF separates the communication from the actual model. Basically BCF introduces a workflow communication capability connected to IFC models (Shafiq et al., 2013). Therefore, BCF does not manage changes in BIM models, while it is a collaborative communication tool only that report these changes through human intervention after found them by the different model checking software (Linhard and Steinmann, 2014).

3.5.3 Using Servers Service:

Some servers for BIM based team collaboration (e.g. Revit server, BIMserver, Autodesk Collaborative Project Management, Drofus, EuroSTEP Server, Graphisoft ArchiCad BIM Server, Horizontal Glue™, etc.…) are developed to act as a central file storage, document and version management tool and as framework for facilitating interaction and collaboration between designer teams on the same project from remote locations over the Internet (Singh et al., 2011). For instance, Graphisoft BIM Server is a server application with ArchiCAD software to exchange, store and manage the shared ArchiCad files based objects of the same project between designers. This server allows managing the selected design changes and showing them in layout revision history window. In this server, the changes in one working session are shared with the others each time when the user save the shared teamwork file (Graphisoft, 2015).

BIMserver software is an open source BIM server (formerly IFC server) from TNO and the University of Eindhoven to explore how collaborative design can be improved through the combination of BIM and open source server technologies (Bimserver, 2015). It is used as a database based on the IFC EXPRESS schema that enables to manage access and trigger remote services, in addition to store, revision, compare, merges and query of different uploaded IFC based BIM (Cahill et al., 2012).

The BIMserver software is written in Java. It uses the model-driven architecture approach to package IFC data into object to ease with which object element queries and filtration can be achieved (Cahill et al., 2012). This means that the software analyses the uploaded IFC file and brought through a process of translation where it is managed into an Eclipse Modelling Framework (EMF) interpretable eCore file. Therefore, the BIMserver does not store IFC files. It maps the single object of IFC in a DB in the server. The core of the BIMserver software is to understand the IFC structure. Therefore, instances of IFC that are uploaded to the model server system are stored on a per-object level. A unique object key is generated for every new instance. This makes it possible to unambiguously identify, retrieve and manage all the information with the IFC model versions (Beetz et al., 2010).

Since BIMserver uses IFC structure, Helm et al. (2010) built the clash detection into the BIMserver through comparing the stored objects in the database between two model versions and filter the collisions based on the Globally Unique Identifier (GUID). Query operations with the BIMserver include the possibility to request all or specific entities (e.g. *ifcdoor*) from the *IfcRoot* entity that identified by a GUID and save the information in Excel file or show them in bimvie.ws. (Bimserver, 2015). An open source BCF server was developed by van Berlo and Krijnen (2014) and integrated with BIMserver through using the traditional BCF in a centralized online setting. The BCF server has shown that project users have been able to create issues, manage them online and evaluate them in context of the actual BIM model.

3.6 Limitations of Managing Design Changes

The current process of comparing and matching two large-scale IFC models to identify changes are time-consuming, cumbersome and tedious. There are some limitations when using the current IFC to manage the design changes. Below are reviewed these limitations.

3.6.1 IFC Standard

The IFC standard is very complicated. There is no direct link between the entities that represent an element and its features. The IFC model developers' main goal is to provide a neutral data format to exchange data among different software programs. It is not developed to deal and manage with occurred changes in BIM models. The IFC neither takes specify a mean for neither sharing the affected changes among different BIM users nor recording the history of earlier changes.

Traditionally, IFC model, which is derived from the BIM model, only reflects the current state of information (Gökçe et al., 2013). It only provides an indication of the change type of the building elements without clarifying the change value. Moreover, change information about element features is not covered in this standard (section 3.3.2.3.1 discussed in detail this limitation). In the vast majority of IFC files, this approach (the change indication) is not functioning to record change type at all. There are no researched that proposes an extension for the IFC EXPRESS schema to manage and version changes in different IFC models. In general, the need for extending the ability of the IFC EXPRESS schema has increased to represent more concepts in different domains, to establish more entities and attributes to allow the IFC model to serve the entire life-cycle of the building and to get an effective integration and collaboration with BIM models.

3.6.2 Current applications

Huge numbers of BIM applications are currently used in the AEC industry. Few of them include some privileges to deal with the design changes. Namini et al. (2011) revealed some of the limitations of BIM applications to identify the flexibility of these applications in applying changes. A questionnaire was distributed among BIM experts about evaluating the ease of applying changes in BIM applications. It was concluded that BIM applications suffer from lacking sufficient artificial intelligence to analyse and manage design changes, and propose alternatives. A set of experiments were conducted by Akcamete et al. (2008) using different BIM applications to assess the ability of these systems for managing changes in BIM models. This paper concluded that the current applications are not able to handle capturing and storing of the history of changes in BIM. There are some limitations that can be summarized below:

- The current trend in some applications that are dealing with the design changes is to compare two design models and detect the differences.
- Most of the BIM applications are proprietary models that need commercial license to use them (e.g. Revit, ArchiCAD, SMC, etc.).
- It is difficult to extend their capabilities because they are not open source code (e.g. ArchiCAD, SMC, etc.).
- Usually, the new versions of the design information are at the model level. None of the applications versions the elements and features to ease managing the changes. Only BIMserver, which versions the model in the IFC object level.
- The users of BIMserver need strong IFC background to understand the comparing and querying functionality.
- Shared (e.g. beams, columns, etc.) and specific (e.g. reinforcement bars, pipes, etc.) changes in the model are not separated to be clarified and sent to the desired recipient.
- Recipients need to do a comparison between the new and old model versions to determine the changes, thus lead to repeat this process by all participants.
- None of the applications deals with the historical information on the models to find the evolution of the elements and their features.

3.6.3 Designers intention

In reality, designers do not want to know that there has been a new model version issued. Whereas, what building elements (e.g. beams column, etc.…) or features (e.g. geometry, locations, etc.…) of the new model version have been changed so that the design changes and the design evolution can be easily extracted and used. The IFC standard and the BIM applications do not include the designer's requirements and demands to deal with the changing information to facilitate his management of the project. It is very complicated for designers to be able to decipher the IFC model in a meaningful way. The designers do not need to know the internal representation of information in the IFC files and the changes in the entity lines, but the meaning of these changes, in terms of changes to elements and features. For instance, it is not necessary to know that the (*IfcIShapeProfileDef*) object in the IFC file has been changed whereas it is worth to know that a specific section value for a particular element has been changed.

3.7 Summary

This chapter studied the BIM interoperability and IFC standard. The specification of the IFC standard of the data schema (IFC EXPRESS schema) and file format (IFC STEP file) are reviewed and discussed. For studying changes in the IFC model later on, the ownership information and the building elements with their features are analyzed and identified from the IFC EXPRESS schema.

Many researches based on IFC STEP file, IFC EXPRESS schema and some IFC domains (central shared model, change management, and versioning) have been outlined. Few of them are dealing with managing the design changes and none of them are proposing an extension for IFC EXPRESS schema to manage changes in different IFC models. Furthermore, few of the BIM applications include some privileges to deal with the design changes. This sets the stage for the next chapter which discusses relevant requirements towards integrating IFC and change management in the building engineering platform.

Chapter 4

Collaboration Versioning Methodology

4.1 Introduction

In the previous chapters, the background on versioning and the preliminary aspects of this research work were discussed. It included reviewing the state of the art in the subject and identifying challenges. In the next chapters, the development process of a proposed methodology is presented. The proposed methodology (Collaboration Versioning methodology) includes two main parts: the proposed development to the IFC standard to integrate the process of versioning with IFC and the development of the prototype software that implements and verifies the developed IFC. The development process of the proposed methodology can be divided into sequential stages containing activities with the intent to achieve better planning and management. The adopted process conforms to the waterfall methodology. This development methodology is a process that flows steadily downwards (like a waterfall) through several phases (Singh et al., 2015). The core activities for the development process of this research work start with identifying the requirements for modelling a collaborative design process using the versioning concept. The next stages include the designing of a collaborative framework and the components that guide the implementation stage of the work. The final stages are the validation and the evaluation. The development process of the proposed methodology is displayed in Figure 4.1, showing the activities consecutively.

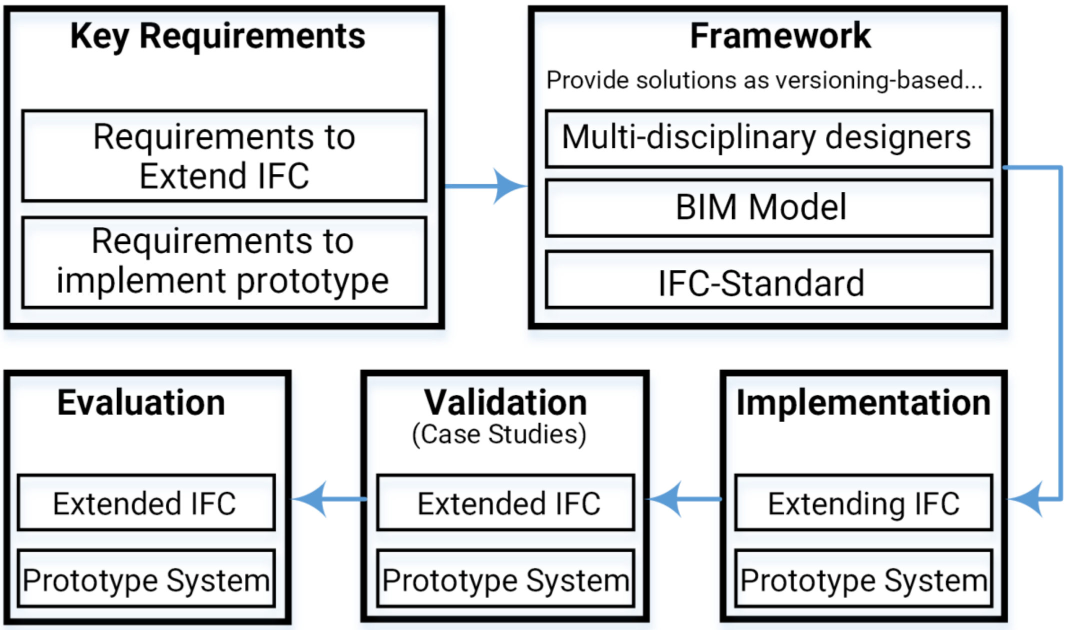


Figure 4.1: The development process of the proposed methodology

This chapter presents the key requirement for adopting the versioning concept (Section 4.2). In addition, it includes designing a collaboration versioning framework to provide a solution through using the versioning concept with the BIM model and among the disciplinary designers (Section 4.3). As mentioned before, the expression “element” has been used in this work to define the building objects (e.g. column, beam, wall, door, etc.…), and the expression “feature” ” has been used to define the information that is related to that building element (e.g. geometry, locations, cost, etc.…).

4.2 Requirements for adopting the versioning concept

From the preliminary investigation stage of this research, presented in the previous chapters, the following key requirements for a proposed collaborative environment were identified. They represent a primitive solution of the end user requirements regarding the proposed methodology. In line with the objectives of this research, the elicitation of the requirements is of two categories: (1) requirements to extend IFC based on the versioning concept and (2) requirements for the prototype implementation. They have both informed the process and development of the collaborative design framework.

4.2.1 Requirements for Extending the IFC standard

Scope of extending the baseline standard of the IFC are the most common research related to IFC (Laakso and Kiviniemi, 2012). The suggested changes to the IFC model need to be readable and understandable among designers to get the interoperability and compatibility among different BIM models. Generally, the extension of the scope of the IFC as a concept might include aspects related to many fields of engineering and different topics. However, the extension of the concept should take into account the existing IFC standard and follow the structure of the schemas and the same inheritance hierarchies. Therefore, the main requirements for extending the IFC to cope with the versioning concept consist of four aspects, as shown in Figure 4.2 and further discussed below:

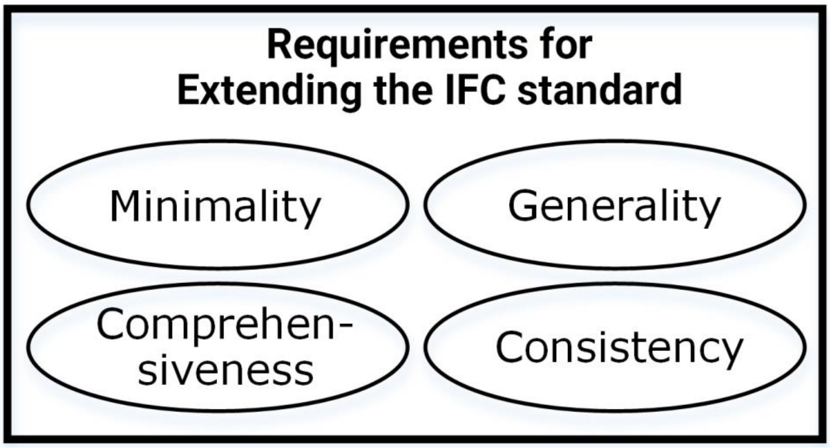


Figure 4.2: Requirements for Extending the IFC standard

4.2.1.1 Minimality

The target is to define the minimum number of new definitions of entities and relationships to deal with the versioning concept that is not within the EXPRESS schema specification and to reuse the available schemas in the IFC EXPRESS. This will avoid the unnecessary expansion of the IFC standard and achieve compactness and better utility. The definitions and data structure of the latest version of available IFC standard (IFC 2x4) are used as the basis for the proposed extension.

4.2.1.2 Comprehensiveness

IFC models tend to be large, complex and with numerous entities. They cover different fields (building, dams, bridges, roads, etc.…) and involve multiple disciplines (Section 3.3.2.1). IFC file contents shares building components between different disciplines in addition to specific components for each of those disciplines. The proposed implementation should cover all the shared and specific elements in the building and involve all the disciplinary teams.

4.2.1.3 Generality

The data schema architecture of the IFC EXPRESS, as presented in chapter 3, is divided into rooted and non-rooted entities. The building elements are within the rooted entities, so that all building elements are following the same inheritance hierarchy derived from the *IfcRoot*. Whereas, the features of an element are within the non-rooted entities, so that each feature has different hierarchy derived from various resources. The proposed schemes that include the versioning concept have to assemble the different resources of the features. They should be generic to be usable and applicable to any change in the building elements and in the features.

4.2.1.4 Consistency

Requirements for the management of different types of changes in the various information models need to take into consideration the activities that take place in all the phases in the life cycle of the building. The proposed versioning system in the IFC standard has to support different change types in the building elements and their features. The proposed entities need to deal with the available information of the model, as well as with missing information. They have to be consistent to handle all sorts of changes. The proposed entities have to, on the one hand, collect the added, modified, and deleted features to the latest model version and, on the other hand, connect all change scenarios for the same feature to record a coherent history of each element feature.

4.2.2 Requirements for implementing the prototype

The core requirements for implementation a software system include coverage of economic and practical aspects and coping with technological development to support the design process in a collaborative environment. Prototype requirements are identified in order to enhance the exploration process that goes into designing the specification of the software system. Efforts were made to achieve a good system and balance of these factors as the research work progressed. The descriptions of various aspects of these requirements are shown in Figure 4.3 and further discussed:

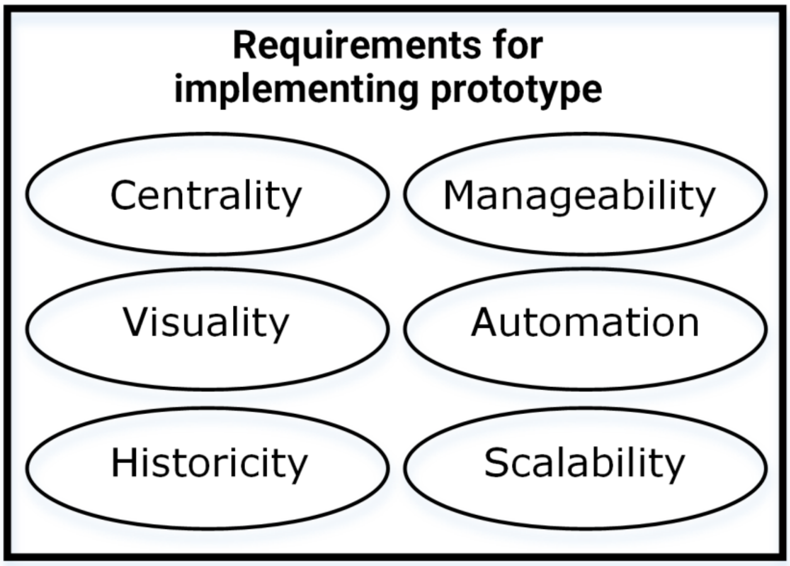


Figure 4.3: Requirements for implementing prototype

4.2.2.1 Centrality

The complexity of each BIM information and the difficulty of exchanging different BIM models among all participants of a project resulted to the provision of a central shared workspace to manage the shared information. However, an IFC model in a shared workspace needs not just to cover the information representation of the building but it also needs to include changes made to building models and their ownership. The role-based access control is needed to be adopted to achieve the access rights of the geographically separated users with appropriate permissions.

4.2.2.2 Scalability

The capability and the performance of the proposed prototype need to accommodate with the growing amount of work without adding new resources to the system. Scalability in this work relevant to the number of the disciplinary teams and models, the sizes of the building, the amount of stored information, the type of changed information, and the number of users of the server.

4.2.2.3 Visuality

A 3D visual model is one of the key requirements for any modern software systems. It is more illustrative than words. The system should enable each designer not only to identify the changes in the model numerically but also to sight the differences between the models graphically. Visualizing changes in a model enables users to understand the affected information, to explore design options, to communicate design intent, and to improve collaboration.

4.2.2.4 Manageability

Manageability within the scope of implementation is a prerequisite in order to obtain the satisfaction of a wide range of users. Successful collaboration is not only about exchanging the information among multi-disciplinary designers; it is also about organising the information to reflect the representation and manipulation of different models, to provide varied user preferences, and to cope with different versioning cases. Managing the changes is one of the main targets of building an effective collaboration. By taking a managed approach in the software system via a change management system, error is minimized, cost is reduced and predicted, time is shortened, and performance is maximized.

4.2.2.5 Automation

The process of comparing models, identifying changes, sharing new information, deleting old information, managing IFC models need to be automated in the proposed system to increase the accuracy of getting the information, reduce the time required to implement the change and minimize human errors.

4.2.2.6 Historicity

The history of the changed information of the building elements from the initial step to the current state is needed to be recorded and embedded within the model to become possible to extract, keep track, and retrieve the historical changes in different versions of the model and at different levels of information.

4.3 Collaboration Versioning Framework

The main requirements for extending the IFC standard and developing of a software system to cope with the versioning concept were identified and discussed in the previous section. These requirements provide a guide to preliminary specifications for designing a framework. The design stage describes how the proposed methodology performs the requirements outlined in the requirements stage.

The goal of this research to manage the design changes is to integrate the process of versioning, as a change management approach; and the use of Building Information Modelling (BIM), as a process to describe the building design. An IFC, which is a data representation standard to exchange information between BIM tools, has been suggested to deal with the versioning concept (Figure 4.4). This Integration introduces an extended model (IFC) for design change management. The relationship among multi-disciplinary designers and the information classification in a single BIM model are two aspects that can be expanded to deal with the versioning concept and to be solutions to implement them in the data model (IFC standard).

Integrating the versioning mechanism within the IFC model develops the capacity of IFC to process dynamic data. This approach combines a design model (current information) with a behavioural model (change information) to enable active coordination of the information. The new process- oriented IFC model covers the current and changes information to provide a foundation for managing changes in BIM and collaborating multi-disciplinary designers.

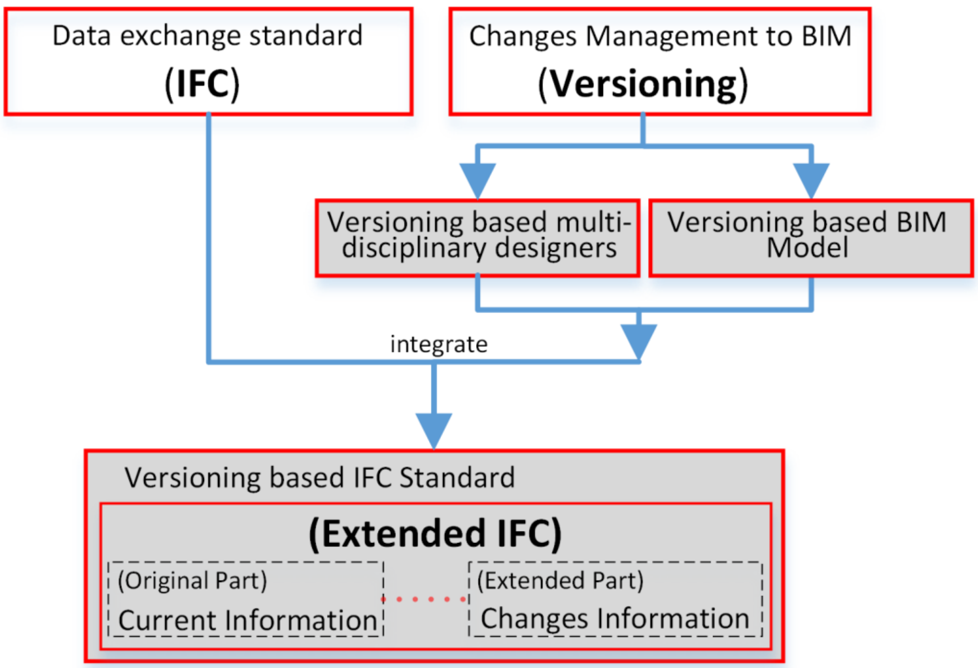


Figure 4.4: Integrating the versioning process within IFC model.

In essence, frameworks provide guidelines. A collaboration versioning framework has been designed to apply the extended IFC in the collaboration design process. The proposed framework consists of three workspaces that are used by the sender, mediator, and recipient. The processes involved in each workspace could be separated into one or more events (modelling, versioning, sharing, etc.…), as shown in Figure 4.5.

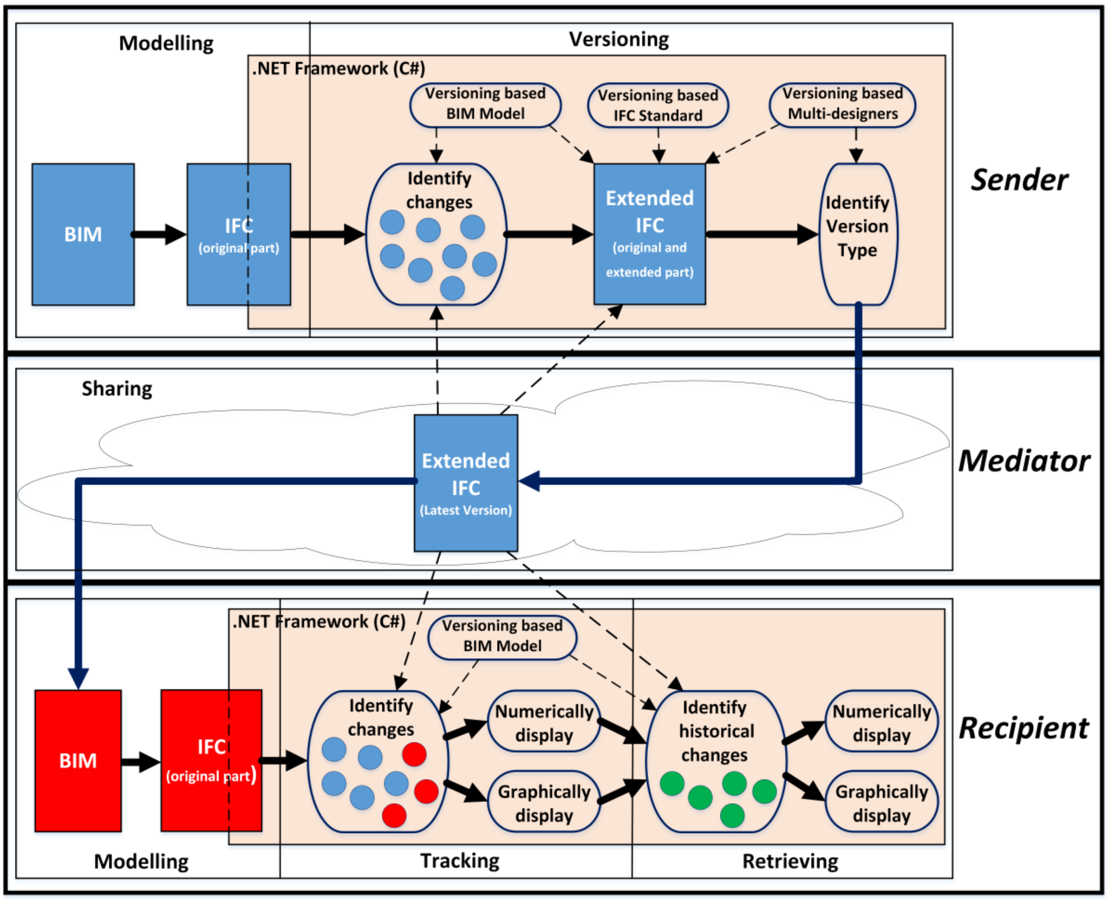


Figure 4.5 proposed collaborative versioning framework

The first workspace is with the sender, who specifies the changes in the information to the other participants. Firstly, the model needs to be transferred from BIM model to IFC model. Secondly, changed information needs to be extracted from the current BIM model and to be added to the IFC model in a standardised format. In addition, all the versioning history should be attached into the IFC model to record all the old information. This Integration between versioning concept and IFC introduces an extended IFC model for design change management.

The second workspace is a mediator, a repository for sharing and exchanging information between the sender and the recipient sides. It is responsible for submitting and centralizing the changes carried out by the sender and for distributing the centralized model among all recipients. The last workspace is the recipients, those who obtain the latest information-oriented versioning from the sender's side. Firstly, the extended IFC model needs to be compared with the current recipient model. Secondly, all current changes need to be extracted and clarified in the recipient's BIM model. Finally, any history information needs to be retrieved and visualized in the recipient's model.

Studying the versioning concept could be increased and classified into two main aspects (as mentioned above). The relationship among multi-disciplinary designers and the information classification in the BIM model have been expanded in the next sections to deal with the concept of the versioning.

4.3.1 Versioning-based Multi-disciplinary designers

The AEC project is complex in nature. It links multi-teams from various domains. During the time of the project, teams generate different BIM models and several processes individually. Changes and modifications are unavoidable even when using the BIM approach. The versioning concept needs to deal with the different BIM models of the multi-disciplinary designers. This section discusses the changes in information that has effect on the disciplinary models and how the changes can be centralized in a shared version among the disciplinary teams.

A large number of disciplines are involved in the collaborative design of any engineering project (architect, structural engineer, mechanical engineer, electrical engineer, sanitary engineer, soil engineer, cost engineer, quantity surveyor, and some other specific disciplines “such as, medical equipment engineer or roads engineer”). For reasons of working within an adequate scope, four of these disciplines (architect, structural engineer, mechanical engineer, and electrical engineer) and their respective BIM models have been represented in the explanations and illustrations of this work.

4.3.1.1 Local and Global Information

For the management of changes in an individual model, it is required to study the effect of each piece of information on all disciplines. The shared building elements (e.g. wall, beam, column, slab, roof, stair, etc.…) are the central and affected components of the raw building used by all disciplinary teams. Designers can make changes to these elements with the consent of others to ensure the consistency of the models. While, the unshared building elements (e.g. structural footing, HVAC ducts, plumbing pipe, etc.…) are specific information related to individual disciplinary domain and changing them does not affect the other disciplines.

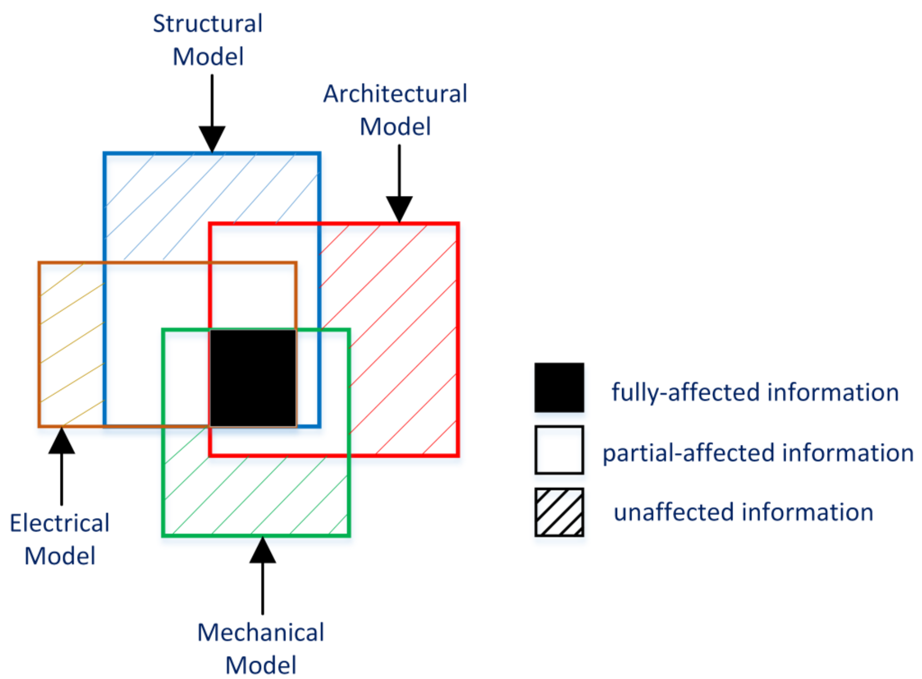


Figure 4.6: Geometric forms portraying different information models

The information models can be portrayed as geometric forms, and interrelated and intersected together as in Figure 4.6. The information in each model can be classified into two types (**affected** and **unaffected** information) and the affected information can be additionally classified into **fully affected** and **partially affected** information. The fully affected information (black area) includes shared information with all the participants in the project, such as columns, beams, slabs, etc.… A change in the information of this type in the model has a direct effect on all disciplines. The second type of the affected information within the model is the partially affected information (white areas). It contains shared information with some participants, such as tiles, basin, duct, etc. A change of this type has effects some of the disciplines. The unaffected type (hash areas) covers the unshared information and changing this type does not have any influence on the other disciplines, such as the structural reinforcement bars.

Thus, changing the affected information is essential for both the inter-disciplinary and intra-disciplinary teams while changing the unaffected information is only essential for the intra-disciplinary teams.

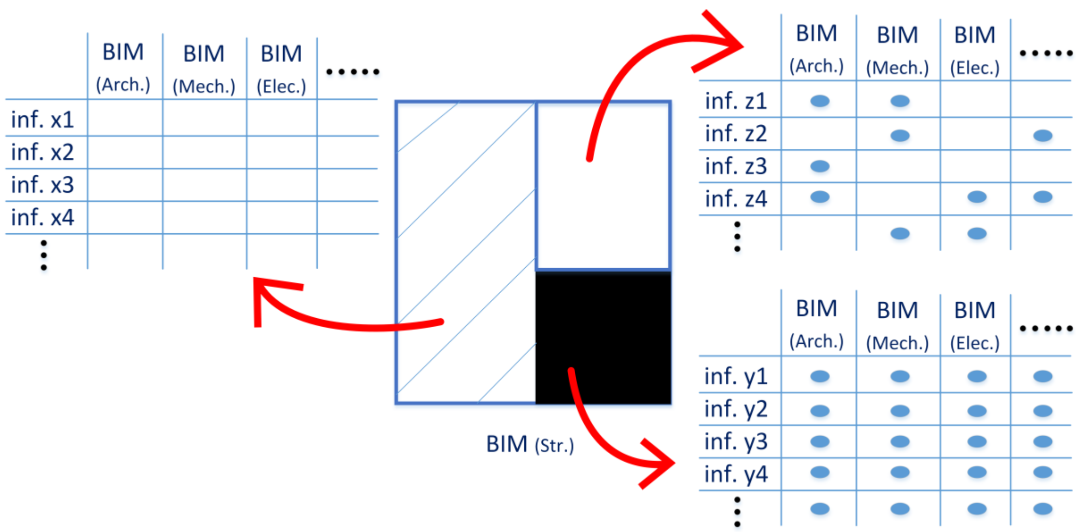


Figure 4.7: The information required in the database

A database change management has proposed to organize the degree of effect of changing elements between multi-disciplinary models. In Figure 4.7, the geometric form that represents the structural model of Figure 4.6 is selected and reorganized to show the three types of information in the database. Information (inf. y_1, y_2 …) represents the fully affected information; information (inf. z_1, z_2 …) represents the partially affected information while information (inf. x_1, x_2 …) represents the unaffected information. According to the necessities of the information based on the discipline, as explained above, the information (y_s and z_s) are essential among inter-disciplinary and intra-disciplinary teams while the (x_s) information is required among intra-disciplinary teams only.

Following from the above premise, two types of versions could be classified depending on the degree of the effect of the changing information on the disciplines.

- **Global Version (G_V):** It is shared among inter- and intra- disciplinary teams, when the changed information in one of the disciplinary model affects all or some disciplinary teams. The affected changes (fully and partially) are under this type of versioning.
- **Local Version (L_V):** It is shared among intra-disciplinary team, when the changed information in one of the disciplinary model affects his disciplinary team only. The unaffected changes are under this type of versioning.

As an example, a set of changed information by one of the structural team is presented in Figure 4.8. The affected and unaffected changes are important for the rest of the intra-disciplinary team (the structural team) while only the affected changes are important for the other disciplinary team. Based on the versioning classification above, the affected changes exist within the global version (G_V) and the unaffected changes are within the local version (L_V). From now on, the expression “global” change/information will be used to represent the “affected” change/information while “local” will be is used to represent the “unaffected” change/information. To demonstrate that on the figure, from the twenty-five pieces of the changed information in the model by the structural team, ten of them are global information and need to be shared with the same and other disciplinary teams in a global version and fifteen changes are local information and need to be shared with the same disciplinary team only in a local version. From these ten pieces, only one of them “a” is shared between all, two of them “b and c” are shared with two teams and the other seven are shared with one team only. As a result, from the global version that is generated by one of the intra-disciplinary team, a set of changed information can be identified for each other intra-disciplinary teams.

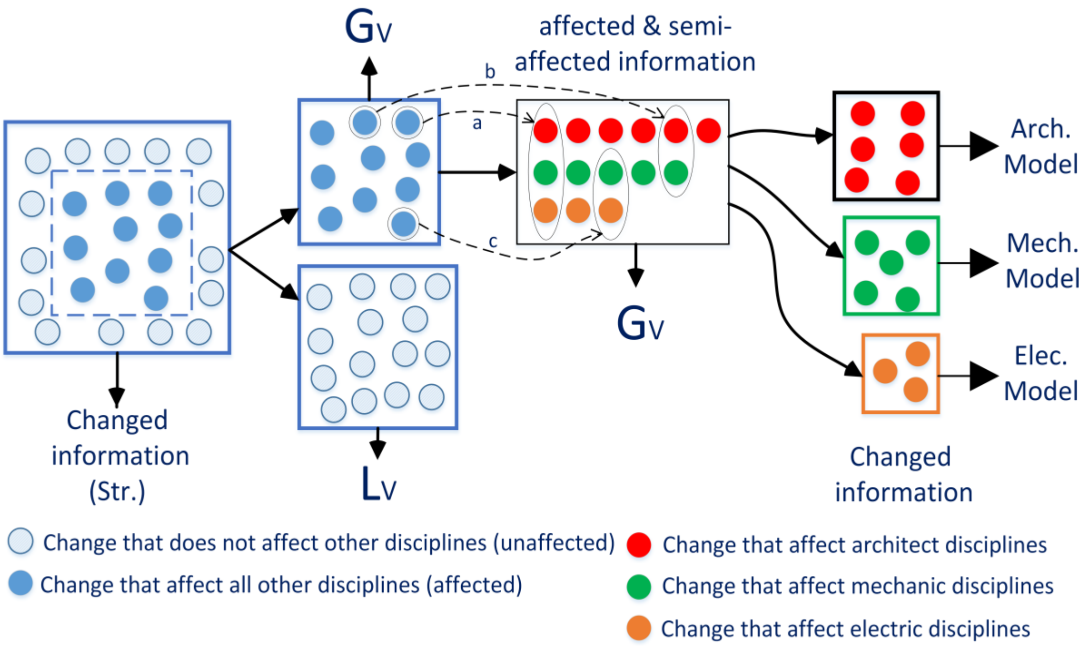


Figure 4.8: Local and Global Versions.

4.3.1.2 Sharing the Changed Information

Each intra-disciplinary team has a central local version shared among their designers and there is a central global version among inter and intra-disciplinary teams. One designer in each intra-disciplinary team (usually the team manager of that team) is also with the team of the inter-disciplinary designers, as shown in Figure 4.9.

The proposed central global and local versions can better meet the requirements of the inter-disciplinary and intra-disciplinary collaboration and provide effective communication modes between the designers from different and at the same disciplines. In this way, real collaboration among inter-disciplinary and intra-disciplinary teams can be achieved through sharing the effected version (G_V or L_V) with the related disciplinary teams.

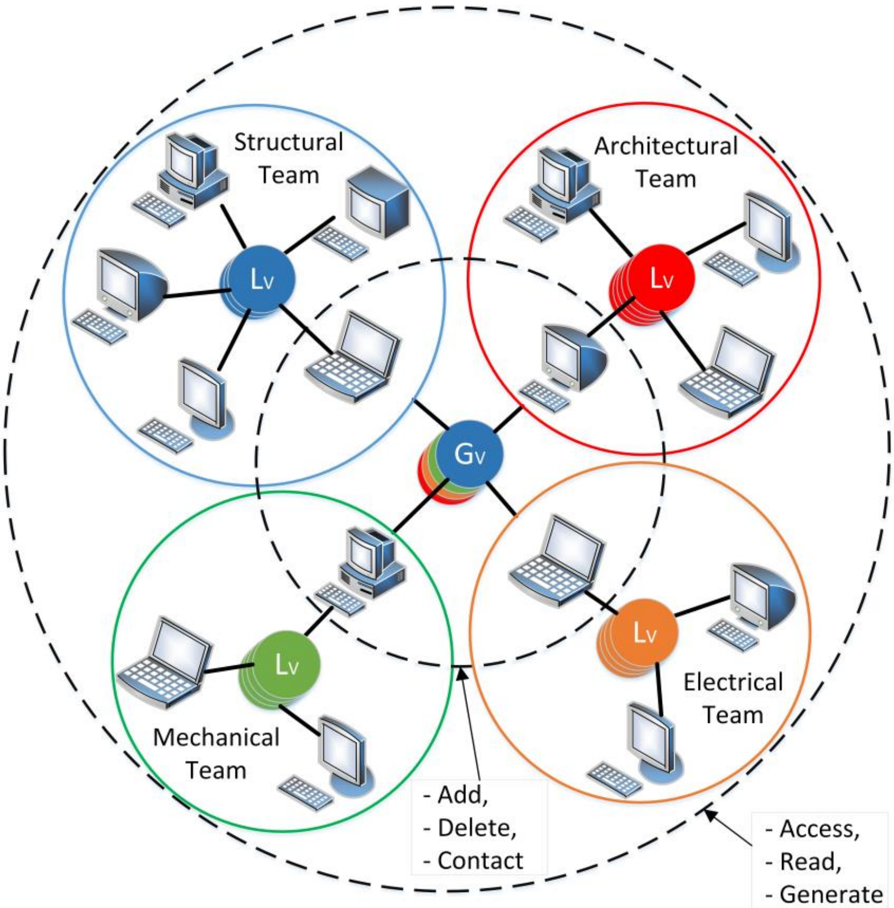


Figure 4.9: Inter-disciplinary and intra-disciplinary teams.

To allow geographically distributed (separated) design teams to share the changed information on a model simultaneously, a Cloud Computing technology was adopted in this study. Multi separate workspaces are proposed in the cloud. Each intra-disciplinary team has their shared local workspace (local cloud) and there is a shared global workspace (global cloud) among inter- and all intra- disciplinary designers. The number of the local workspaces includes the overall participants of the multi-disciplinary teams in the project. Each local cloud is private for the participants of that intra-disciplinary team but the global cloud is public with some restricts for the intra- and inter-disciplinary teams. All the participants in each local cloud have the right to access the workspace, to generate new local version file and share it with the same intra-disciplines, and to read or delete the current local version file. On the other hand, all the participants in the global cloud (as demonstrated in Figure 4.9) have the right to access the global workspace, to read the current file and to generate new global version file but not to share it with the other intra-disciplinary teams or delete the current file. To better manage the collaborative design process, only inter- disciplinary participants have the right to share the new global file, delete the old file, and make a contact with the other intra-disciplinary teams. Table 6.1 shows the different levels of permission available to the designers to use the global cloud.

Table 6.1 Using global cloud by different designers

Global Cloud	Access/ cloud	Read / file	Generate / file	Add/ file	Delete/ file	Contact /other teams
Inter-disciplinary designer	√	√	√	√	√	√
Intra-disciplinary designer	√	√	√	×	×	×

The proposed multi separate workspaces in the cloud support collaboration among designers that are all working on their models in parallel and sharing a new design version as a central model in the workspaces. Many systems of collaboration mode (synchronous and asynchronous collaboration) are used among designers that depends on the ability to (access, open, read, modify, generate and delete) the shared model (Yao et al., 1999). These systems are ranged between pessimistic to optimistic approaches. The proposed approach is more optimistic that allows access and management of the shared file with varying degrees based on the above permissions. Figure 4.10 illustrates the workspaces in the cloud. It shows that only one designer in each intra-disciplinary team has the permission to send the new global version file to the global cloud and share it with the whole disciplines.

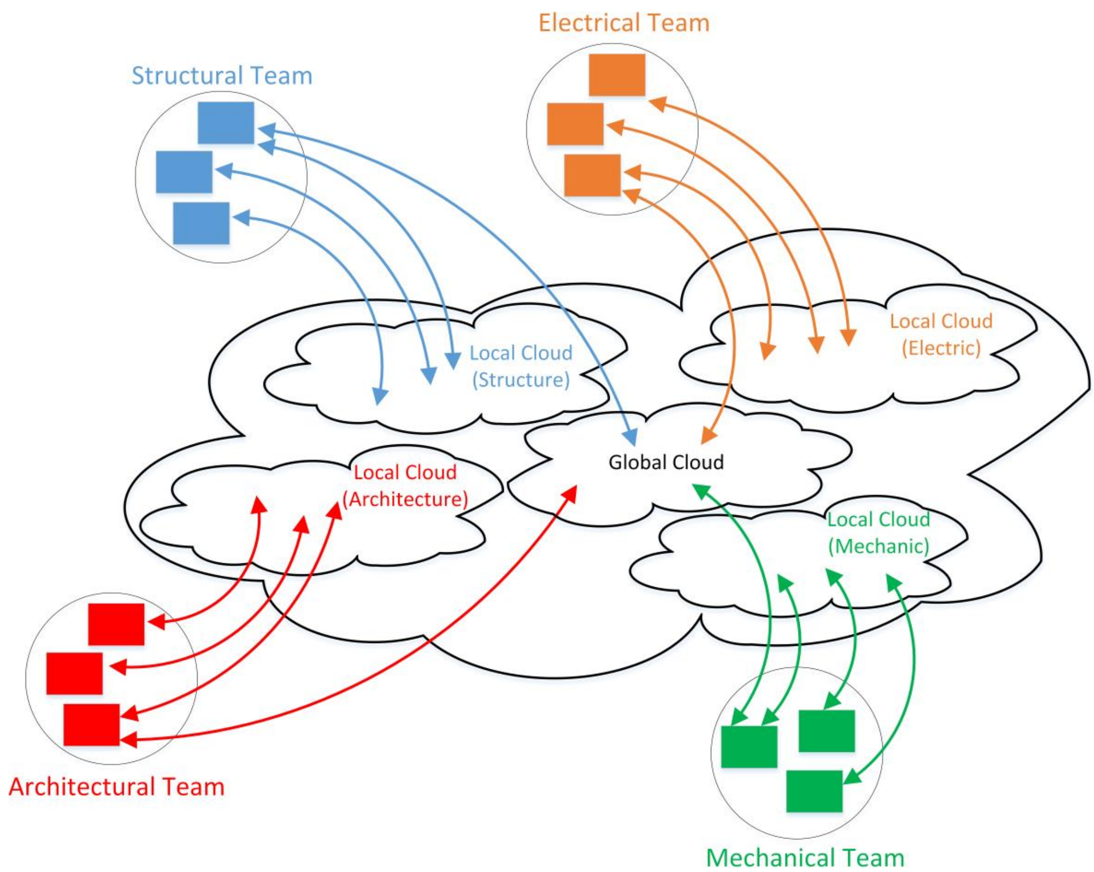


Figure 4.10: Sharing the new file in the Global and Local Cloud

The local and global clouds are independent of each other. Generating a new version in one of the workspaces is not necessary to generating another new version of the model in the other workspaces. A new model version is generated based on the degree of the effect of the changes (local or global changes). These changes will be incorporated in the new model version (this point will be discussed more in chapter 5 and 6).

A versioning scenario has been proposed in

Figure 4.11 between a set of local versions for one of the intra-disciplinary designers (structural team L_{SV}) and a set of global versions (G_V). For example, V_3 and V_{S4} are generated at the same time because the changes in the model were sufficient to generate a global version for the global "affected" changes and a local version for the local "unaffected" changes. Whereas, V_2 is generated only at the time of the changes in the model carry information on the global changes that were enough to generate only a global version.

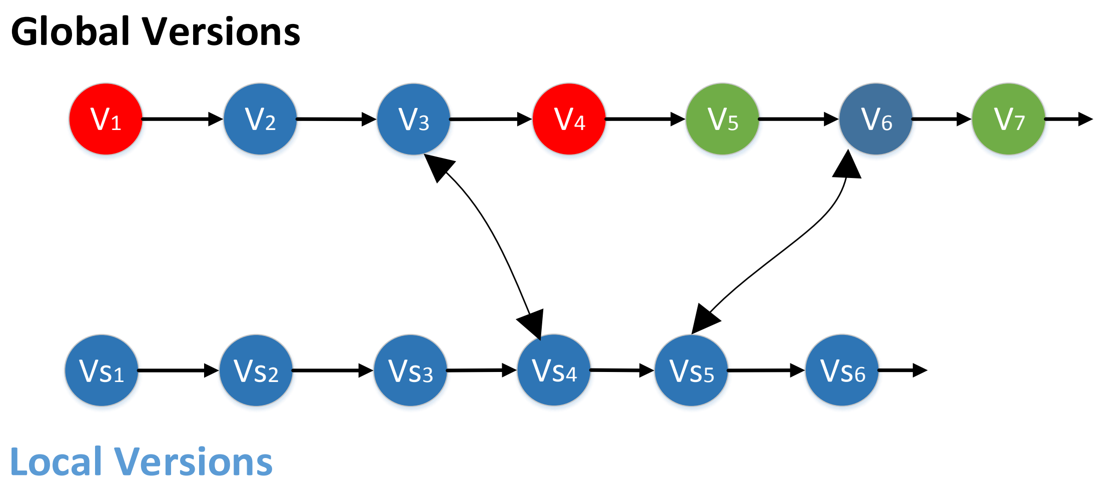


Figure 4.11: Proposed versioning process

4.3.2 Versioning-based BIM model

BIM provides intelligent, data-rich digital objects. Any object in the BIM and any property for each object can be defined and managed (Eastman et al., 2011b). In order to certify the collaboration of multi-disciplinary BIM models to support model changes, versioning is being used in the BIM model to manage the changes in the building elements and their features. Any design version represents the state of development at a particular time. A new version can be generated by identifying at least a single difference in comparison with a predecessor version. This may include the difference in a feature of an element.

This section addresses concepts related to the structure of the building information in the model to generate an information version and information change that can effectively manage all changes in the BIM model during a design cycle. Versioning the information in the project has been classified into three gradual levels from the largest to the smallest. It is starting from the whole model information, then the information in the element and then the information at a very narrower level, the feature. The levelling approach is intended to make the information in the model easier to manage.

4.3.2.1 Model Versioning (M_V)

Each new version of the model, as the design progress, represents a set of information required for the design and construction of a project wherein a given design activity is carried out. The conventional BIM model does not allow identifying the information that has been changed and/or the accumulated information that has been added to the model since the last state. The new model version is like a snapshot, providing information of the state of the design process at a given moment.

Figure 4.12 illustrates the current state model that might contain a set of unchanged and changed information. Each circle in Figure 4.12 represents one set of information. As mentioned in chapter two, there are three different types of changes that could occur since the last model version (add, modify, and delete). The deleted information is missing in the current model. The proposed model for versioning includes the deleted together with the added and modified information to make up all the change sets that requires being included within the current model.

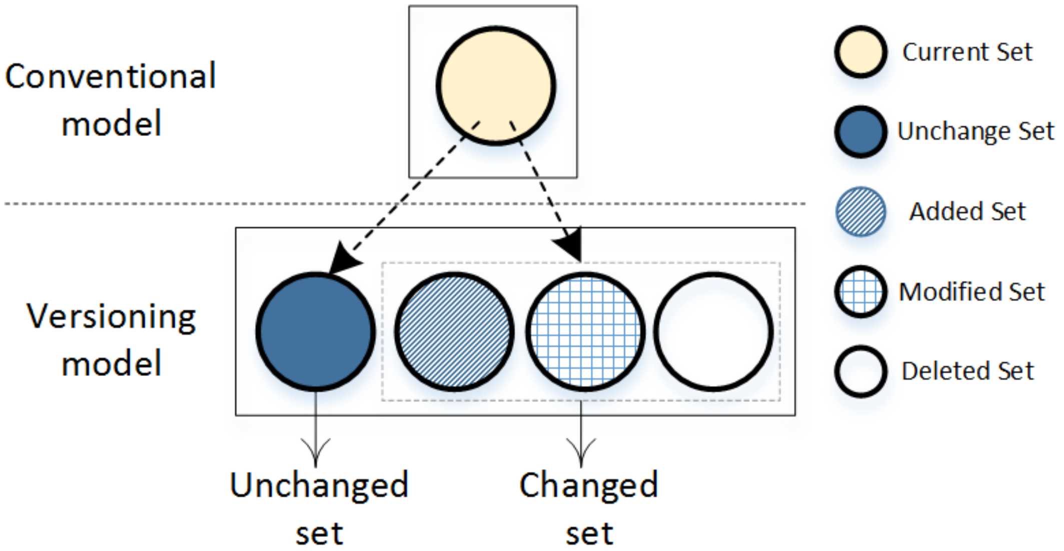


Figure 4.12: Conventional and versioning models

The model version is defined as a "capital M" with an index representing the version number of this model (e.g. M_0). The index "C" with "capital M" and version number represent all changed information in the model version (e.g. M_{0C}). An illustrative simple prototype BIM model (M_0) is shown in Figure 4.13. It consists of four columns and four beams. With the progress of the project, a series of model versions ($M_{0 \ldots}$ M_n) can be generated by any disciplines to represent the up-to-date state of the model at a particular time. The rectangular shapes represent the model versions whereas the circular shapes represent the change versions. In this example, there are four-model versions (the prototype "M_0" with three-updated models "M_1, M_2, and M_3"). The change versions (M_{0C}, M_{1C}, M_{2C}, and M_{3C}) for the model versions are generated to demonstrate the changes in the elements and features. The next sections discuss versioning of elements and features in more details.

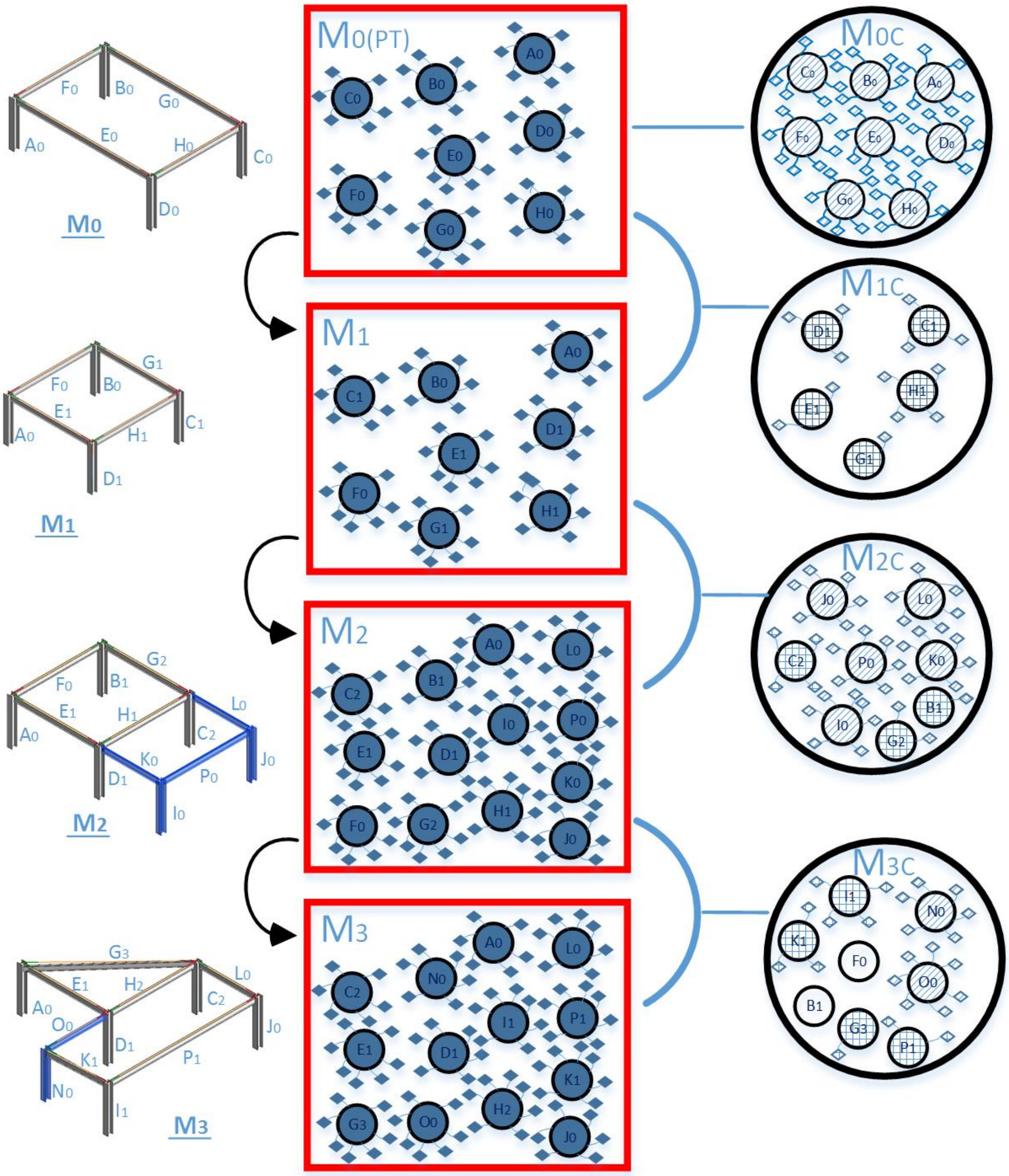


Figure 4.13: example for creating model and element versioning.

4.3.2.2 Element Versioning (E_V)

Since BIM involves representing a design as combinations of objects (Oti and Tizani, 2015, Nawari, 2015), it is possible to manage the information in the model at the current state of the design process down to the building elements level. A single building element is a compilation of a number of features. It represents a repository of the collection of all its features. Each building element in the new model version can be versioned based on the changes in the features. Therefore, changing a feature of an element means a new version of that element is saved in the model.

From the model versions ($M_{0, \ldots,}$ M_3) of Figure 4.13, a set of elements are described that are defined as a "capital letter" with an index representing the version number of this element.

$M_0 = \{A_0, B_0, C_0, D_0, E_0, F_0, G_0, H_0\}$

$M_1 = \{A_0, B_0, C_1, D_1, E_1, F_0, G_1, H_1\}$

$M_2 = \{A_0, B_1, C_2, D_1, E_1, F_0, G_2, H_1, I_0, J_0, K_0, L_0, P_0\}$

$M_3 = \{A_0, C_2, D_1, E_1, G_3, H_2, I_1, J_0, K_1, L_0, P_1, N_0, O_0\}$

A series of changes in the model versions are identified and extracted at the element level to describe the revision between the earlier model version (M_{n-1}) with the new model version (M_n). In each new model version, changes in the elements in the previous example incorporate the modification in some existing elements (e.g. $C_{1,}$ $D_{1\ldots}$) and the addition in some new elements (e.g. $I_{0,}$ $N_{0\ldots}$), but not the deletion of elements (e.g. $B_{1,}$ F_0). This enable to generate a change set (M_C) for each model version at the element level to represent the entire changes (add, modify, and delete) of the elements. As represented below.

$M_{0C} = \{A_0, B_0, C_0, D_0, E_0, F_0, G_0, H_0\}$

$M_{1C} = \{C_1, D_1, E_1, G_1, H_1\}$

$M_{2C} = \{B_1, C_2, G_2, I_0, J_0, K_0, L_0, P_0\}$

$M_{3C} = \{\underline{B}_1, \underline{F}_0, G_3, I_1, K_1, P_1, N_0, O_0\}$

M_{C3}, for example, represents the eight changed elements (two new, two deleted and four modified elements), and so on for the other model changing.

4.3.2.3 Feature Versioning (F_V)

The information of each element in the same BIM model differs from others in terms of the features that represent the element. Element features in the BIM come from a variety of sources. It covers more than just geometrical part of the element. It includes material, specification, quantities, analytical parts, and may have some extra information about cost and sustainability.

These features can be classified in BIM depending on the necessity for having the information and the availability of its value, into two types: the first type is permanent, it is essential feature that must be defined with the element, but its value could be changed in each model version (like the shape, location, material, etc.… of the element). The second type is temporary, it is optional feature that could be added or deleted several times with the maturity of the model, and its value is changeable in the model versions (like the quantity, cost, etc.… of the element). Therefore, the features of the elements in a model can be represented as an open matrix with variable numbers of elements for each model version and variable number of features for each element version (Figure 4.14). For instance, the number of column features is different from the slab features, and even the numbers of columns features are different from each other because of the temporary features within the element.

Any change in a feature represents a change in the overall BIM model. Element's features based versioning is not available in today's end-user software. With the intention of implementing a comparison algorithm, feature versioning has been developed in this work. It has been used to manage changes across several discipline-specific models and to make it possible to have multiple versions at the features level.

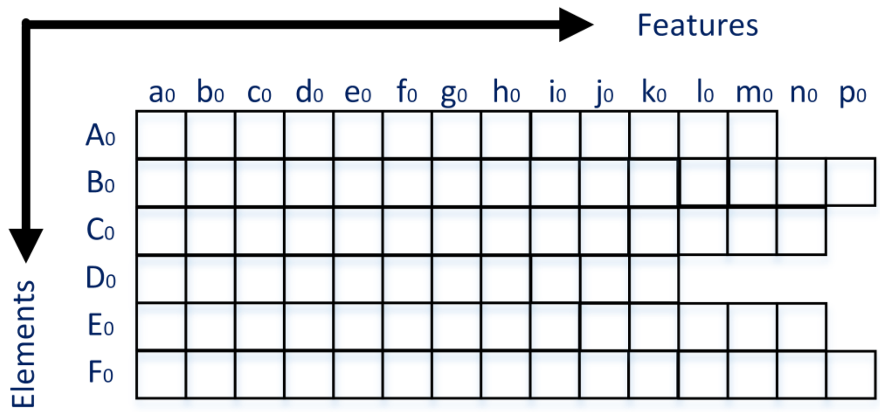


Figure 4.14: Open matrix between the elements and the features

The same indication for the set of elements in the previous section can be applied to the features. "Lower case" letters are used to represent the features of each element with an index representing their versions, as the features of the added element (N_0), and deleted element (B_1), and the features of the element (G_2) that has modified to a new version (G_3) in the model version (M_3) in Figure 4.13.

$N_0 = \{a_0, b_0, c_0, d_0, e_0, f_0\}$

$B_1 = \{a_0, b_0, c_0, d_1, e_0, f_0, g_1\}$

$G_2 = \{a_2, b_0, c_0, d_1, e_0, i_0, g_0, h_0\}$

$G_3 = \{a_3, b_0, c_1, d_1, g_0, h_0, j_0\}$

The features of a single element are represented in Figure 4.15. A circle shape represents an element, while a diamond shape refers to a feature.

Available BIM tools represent the current situation of the element, as in Figure 4.15(a), without clarifying the new or modified features or including the deleted features or the old information about the modified features. The features information needs to be easily identified by the designers, so they can receive full information on the changes in the new model version. The ideal clarification of the features information within an element is shown in Figure 4.15(b) with identification for all new, deleted, modified, and unrevised features information in the new model version. Therefore, change in features can be classified based on carrying information within the model version into available and missing changes. The next two sections discuss these two types of changes.

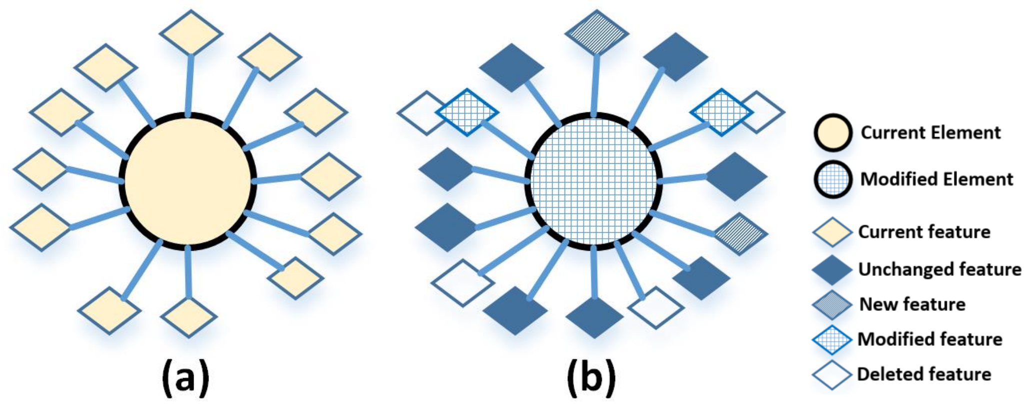


Figure 4.15: Representation of the features for a single element.

4.3.2.3.1 Available Changes (AC)

All the added and modified information in the latest model version can be classified as available changes within the current state of the model. The feature information for each element is used to extract the current changes at the feature level. The modified feature “c_1”, and the new feature “i_0” are indicated as changes in the information of the current element “G_3“in the model version “M_3“. The available change of the modified element “G_3“is referred to as G_{3AC}.

$G_{3AC} = \{a_3, c_1, j_0\}$

Features of the deleted elements (e.g. B_1 in M_3) do not exist in the available changes while all the features of the new elements (e.g. N_0 in M_3) are in the set of the available changes.

$B_{1AC} = \{\ \}$

$N_{0AC} = \{a_0, b_0, c_0, d_0, e_0, f_0\} = N_0$

4.3.2.3.2 Missing Changes (MC)

All the deleted information in the latest model version can be classified as missing changes within the current state model. The deleted information includes the modified features of the earlier version and the deleted features. The feature information for each element is used to extract the missing changes. The modified feature "c_0" and the deleted feature "e_0" in the element "G_3" are signified as missing information at the feature level. The missing changes of G_3 are referred as G_{3MC}.

$G_{3MC} = \{a_2, c_0, i_0\}$

The features of the deleted element B_1 is unavailable information in the new model version and it is required to be documented as missing information whereas the features of the new element N_0 are all exist in the new model version.

$B_{1MC} = \{a_0, b_0, c_0, d_1, e_0, f_0, g_1\} = B_1$

$N_{0CC} = \{\ \}$

To deal efficiently with all changed information, a change set of each element version is proposed to represent all added, modified, and deleted features. It includes jointly the information of the available and missing changes (as in Figure 4.16). The element for G_2, B_1 and N_0 are:

$G_{3C} = G_{3AC} + G_{3MC} = \{a_3, c_1, j_0\} + \{a_2, c_0, i_0\} = \{a_2, a_3, c_0, c_1, j_0, i_0\}$

$B_{1C} = B_{1MC} = \{a_0, b_0, c_0, d_1, e_0, f_0, g_1\} = B_1$

$N_{0C} = N_{0AC} = \{a_0, b_0, c_0, d_0, e_0, f_0\} = N_0$

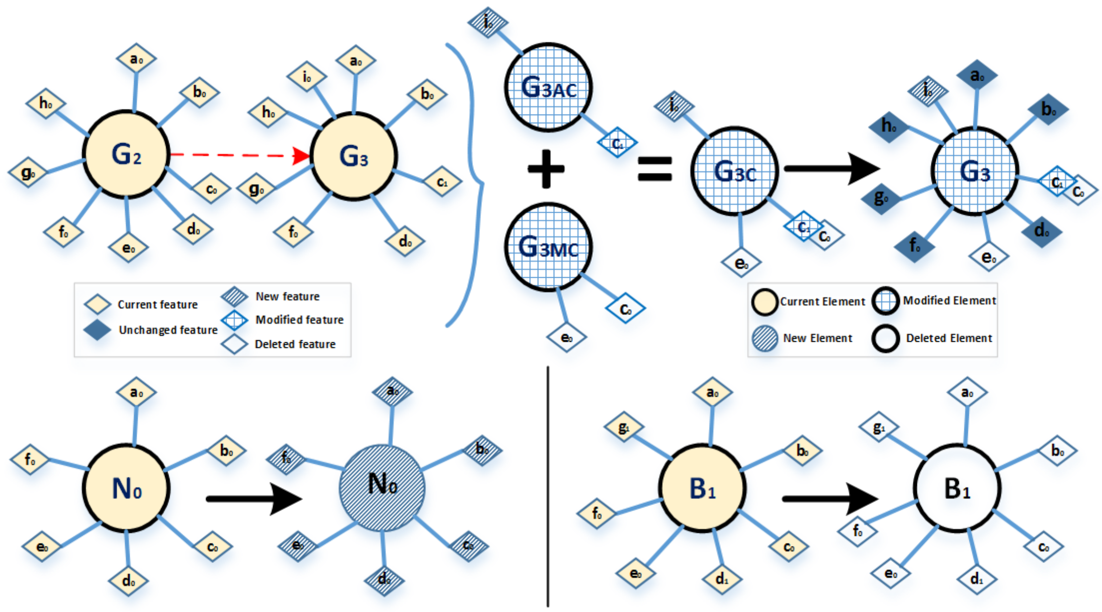


Figure 4.16: The proposed (added, modified, and deleted) features

The changes in the modified elements include part of the element features while the changes in the added and deleted elements include the whole features. Based on that, managing the modified elements can be done at the feature level while managing the added and deleted elements can be done at the element level. Table 4.1 illustrates the probability of the presence of changed information in the current model.

Table 4.1: Presence of the changed information in the current model

Element, Feature		**Available Changes**	**Missing Changes**
Added Element		√	
Modified Element	Added Feature	√	
	Modified Feature	√	√
	Deleted Feature		√
Deleted Element			√

4.3.2.4 Evolution Graph

The typical graphical representation of versioning (version graph) has been used and expanded to store the history of different versions and to study the logical sequence of the evolution of the information. The current version graph includes only the model and element versions in a way that all the available changed and unchanged information in the elements are exist.

Since the model is mainly a set of elements with associated features, the version graph has been expanded to include the versioning information at the level of the features beside the versioning information at the level of the elements and models. The Model Evolution Graph has been suggested to interrelate different model versions (M_V), element versions (E_V) and feature versions (F_V) with each other. Due to the complexity to represent the whole elements of Figure 4.13, three elements have been used (G, B, and N) to show the modified, deleted, and added elements respectively. The model evolution graph is presented in Figure 4.17.

Each graph consists primarily of a set of nodes and a set of arrows that are interrelating with the nodes. The node in this work is formed as a rectangle, circle or diamond shape to represent a version of the model, element or feature respectively. Various forms of arrows were used to link the different version types. Any version in the graph represents the current state at this stage of development. It can be noticed that the version indices for the model can be different from those of individual elements and features. For example, at version (2) of the model (M), the version of element (B) is (1) and the version of feature (a) is (0). Therefore, the same feature version can be in several element versions and the same element version can be in a number of different model versions.

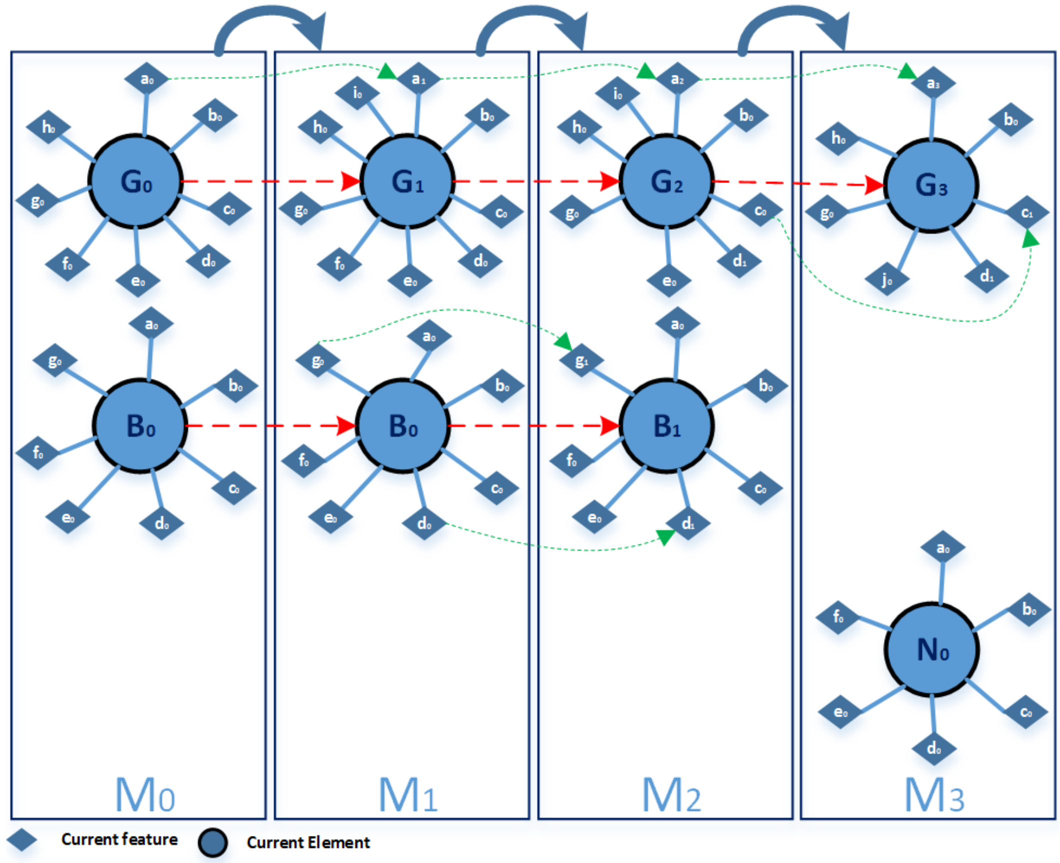


Figure 4.17: Model Evolution Graph

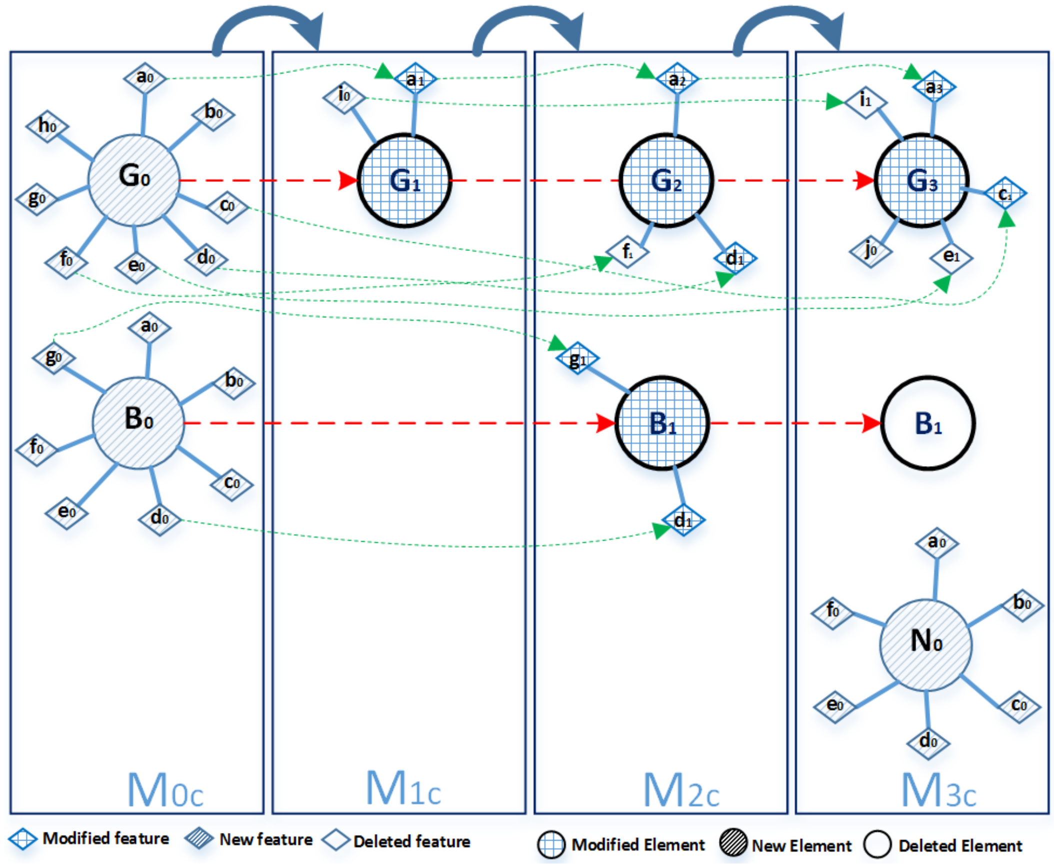


Figure 4.18: Change evolution graph

Changes can be distinguished from the model evolution graph to develop a new graph “Change Evolution Graph” for the changed information only at different levels. This is shown in Figure 4.18. The graph provides the capability to structure the progress in the changed information at different times without redundant and repeated information, and to clarify a full account of the changing information through the inclusion of the missing information (the deleted elements and features). Therefore, various versions of the changed elements, and features are displayed, including the missing element (B_1) and features (i_1, e_1).

4.3.2.5 History Versioning (Hv)

The model, element, and feature versioning in the Change Evolution Graph would allow the full model of any previous version to be reconstructed and tracked historical information for any element. In general, versioning is formed linearly to show the evolution of the information (Taentzer et al., 2014). The change evolution graph for the modified element (G) in Figure 4.18 has been reformed in a circular way to gather all changes taking place at different times around the element, as in Figure 4.19.

The new form represents the “History Versioning” for all information that was changed in an element. The history versioning provides two main benefits. They are:

- Review the changes history of any design element. Each element records its own changes history since its creation. The history versioning provides the ability to filter and track the changes made upon the element.
- Retrieve the changes of any design element. Full information for any model version of the elements can be reformed and rebuilt from the

history versioning through collection of the changed information from the related versions, and retrieve all the features information.

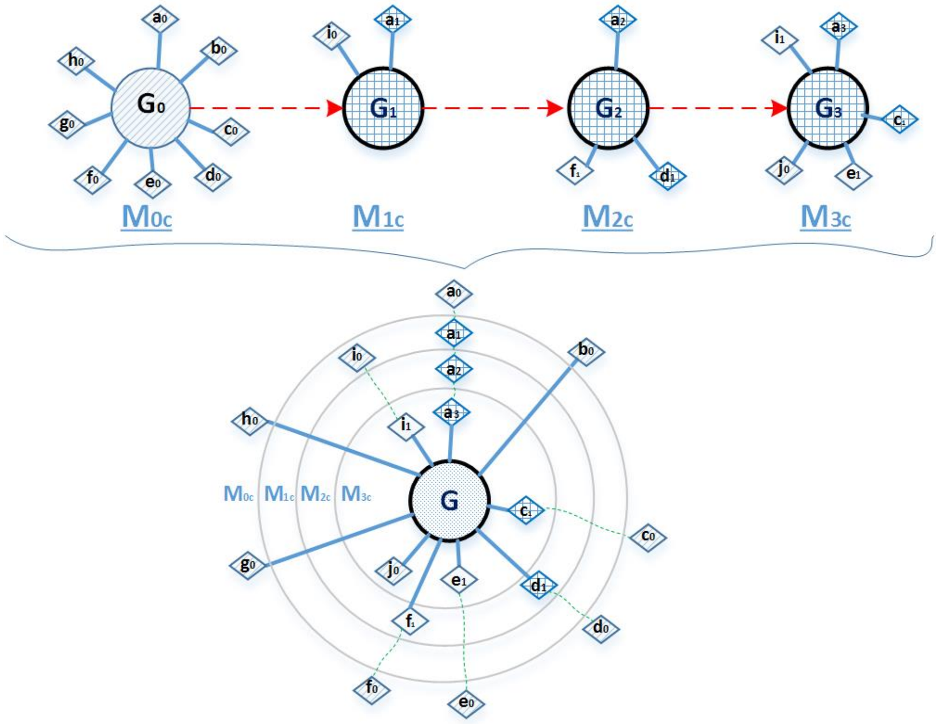


Figure 4.19 The linear and circular versioning for a single element

In the history versioning, the development of an element at different model versions in terms of adding new information and, modifying or deleting existing information can be identified from the versioning information of each feature, which is shown along a sector line. The full sector line represents the latest versioning information and the dotted sector line represents the links between the versioning information of each feature. Each ring describes the changes of the model at that stage (e.g. M_{0C} to M_{3C}). The outer ring is the base model (M_{0C}).

As an example of regenerating and retrieving an element to any earlier version, element “G” is brought to version (0, 1, and 2) and the current version (3) is reformed (as G_0, G_1, G_2, and G_3 in Figure 4.17). Figure 4.20 shows the retrieving process. The features that are new or modified at the required version are kept as they are while the information about the other features is collected from the earlier versions (starting from the newer to the base version). Therefore, to regenerate element "G_2", features (a_2 and d_1) from M_{2C}, (i_0) from M_{1C} and (b_0, c_0, e_0, g_0, h_0) from M_{0C} are collected to represent full information about element "G" at model version 2. This information represents changed and unchanged information at the required state of the model. The same process can be used to reform full information for any model version by regenerating the whole elements at that model version.

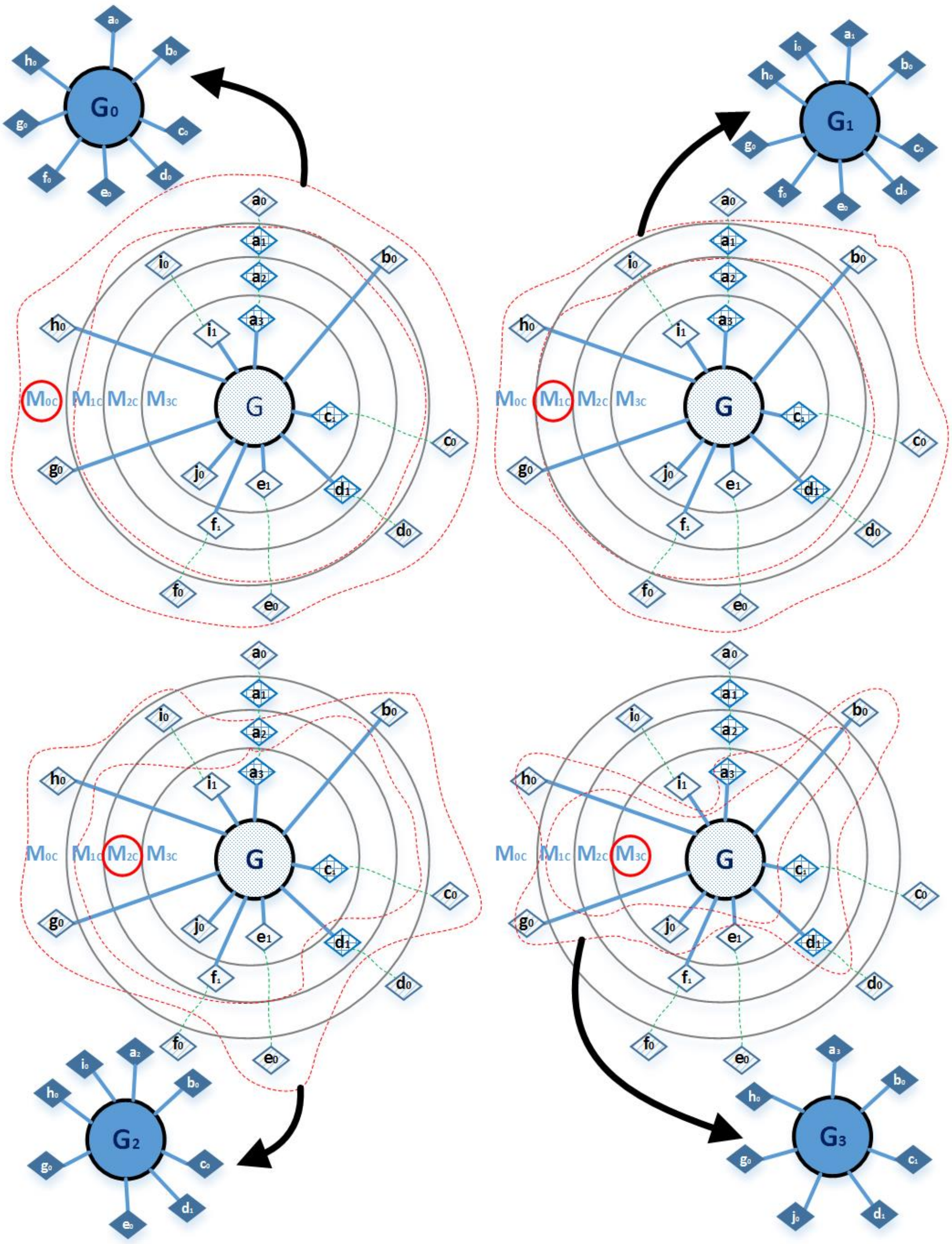


Figure 4.20: Reconfiguration element "G" for all preceding versions.

4.3.2.6 Proposed Model Version

A full scale of the proposed model version is presented in Figure 4.21 to show the current model version and to clarify all versioning types demonstrated in this study. Each single shape (a circle "element" with a set of diamonds "features") represents a separate element version at a particular time. Two new, two modified and two deleted elements among a set of elements are presented as six elements that have been changed (E_C). The two deleted elements are presented as missing information (E_{MC}) while the two new elements are identified as available information (E_{AC}). Modified elements are generated through merging the missing with the available information together. Both are complementing each other, the new with the old modified features are linked together and collected with the new, deleted and unchanged features to form the modified element.

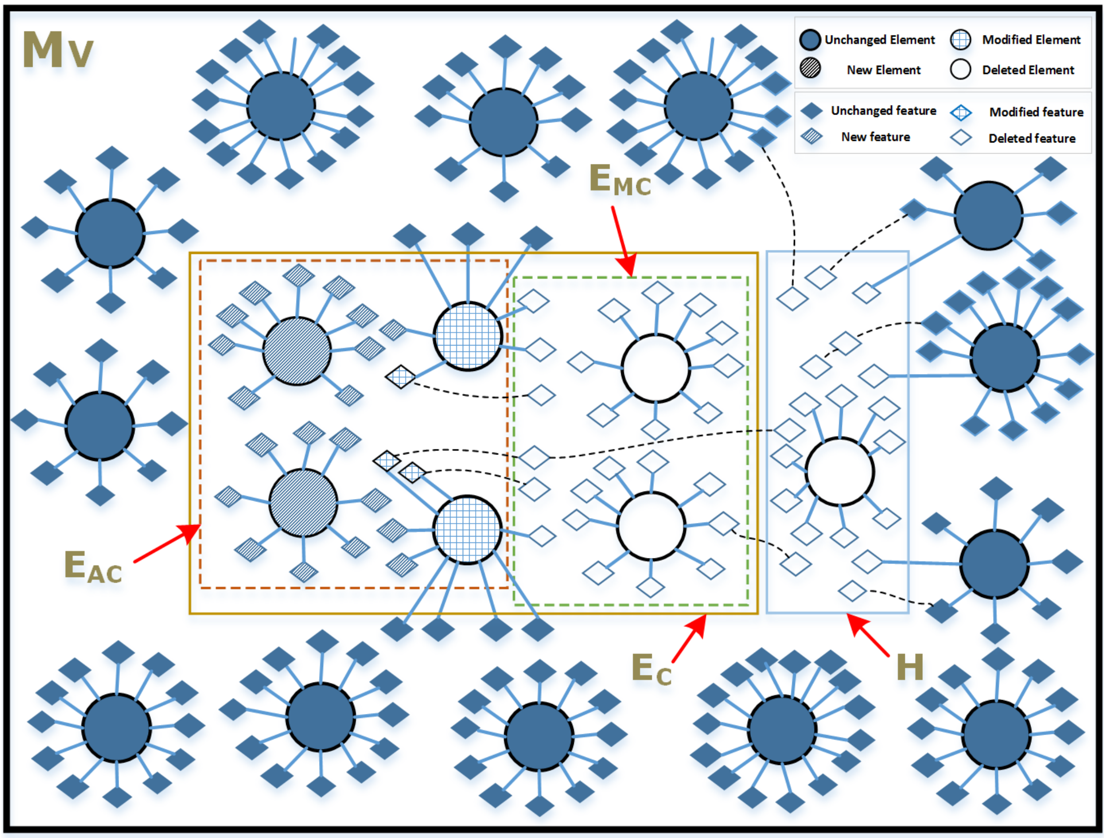


Figure 4.21: the proposed model version

The earlier information within the current model version is represented as "H" to show all the changes in the previous model versions. The earlier information is the missing information in the current model version that includes the deleted features, the old values of the modified features and the deleted elements.

4.4 Summary

The versioning concept is the main aspect in this work. This can be used to develop proposed collaboration versioning methodology. The proposed methodology comprises two main parts (development of the IFC standard with a versioning extension and development of prototype software). Based on that, two sets of requirements were identified to formalise a collaborative framework. The first set is the requirements for extending the IFC standard in a manner that is Minimality, comprehensiveness, generality, and consistency to the extending the IFC standard. The second set is the requirements of implementing collaboration versioning prototype that encompass centrality, scalability, visuality, manageability, automation, and historicity. These requirements guided the development of the collaboration versioning framework.

The components of the proposed framework were discussed in this chapter. The first component covered studying the changing information that has an effect on the disciplinary models and how can centralize the changes in a shared version among the disciplinary teams. Three types of affected information to the designers (fully affected, partially affected, and unaffected information), two types of versioning among designers (global and local versions) and two types of sharing workspaces among multi-disciplinary designers (global and local clouds) have been suggested to cope with the different disciplines and models in the project.

The second component of the proposed collaborative framework used the versioning concept in BIM model to manage the changes in the building information in three gradual levels (model, element, and features). The proposed information model for versioning includes the deleted information together with the added and modified information to structure all the changes set that requires being included within the current model. The current version graph has been expanded to study the versioning evolution at the features level beside the model and element levels and a new version graph has been developed to study the changing information only at different levels. A version history of the features of each element is proposed to review and retrieve any change of the element features.

Chapter 5

IFC – based Implementation of the Versioning Concept

5.1 Introduction

In the previous chapter, the first two stages of the development process of the collaboration versioning methodology were described. The first stage was to identify two combinations of the requirements for modelling the collaborative design process using the versioning concept. The first was the requirements for extending the IFC standard and the second was the requirement for implementing the prototype. These requirements guided the second stage, which entails defining and designing the collaboration versioning framework. The provided solutions to build the proposed collaborative versioning method can be summarized into the following points:

- The proposed system consists of three players (sender, mediator, and recipient) and different events (modelling, versioning, sharing, etc.…).

- Two types of versioning (global G_V and local L_V versioning) are proposed according to the degree of effect of the change of information for different disciplines.
- Each intra-disciplinary team has a central local version (L_V) shared among their designers and there is a central global version (G_V) shared among the inter- and intra-disciplinary teams.
- Cloud Computing technology was suggested to allow the sharing of a single model through a server for each versioning type (G_V and L_V).
- Building information has been classified into three versioning levels: the whole model information (M_V), the element information (E_V) and the feature information (F_V).
- Change in features can be classified based on saving of information within the model version into (available change (AC) and missing change (MC)).
- The proposed information model for versioning includes the deleted information, as well as the added and modified information based for all three-versioning levels (M_V, E_V and F_V).
- Version evolution graph are expanded to study the versioning at the feature level beside the model and element levels.
- Change evolution graph are suggested to structure the progress in the changed information.
- A version history (Hv) of the features for each element in different models is proposed to review and retrieve the changed information.

The requirements and the design stages serve to build of the next stage of the development process, which is the implementation stage. Implementation is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy (Šilingas and Butleris, 2015). As mentioned before, the proposed collaboration versioning methodology includes two aspects: the IFC standard and the prototype software. Therefore, implementation of the proposed method can be divided into two main parts: Implementation through extending the IFC standard and through developing prototype software. In this chapter, extending the IFC standard is covered to provide details on integrating the process involved in the versioning concept into the IFC standard.

5.2 Versioning-based IFC-Standard

The solutions that have been provided based on multi-disciplinary designers (Section 4.3.1) and information classification of the model (Section 4.3.2) to deal with the versioning concept were implemented in the IFC model. The standard has been extended in this section to include the concept of versions. The implementation of the IFC extension is presented in two parts. The first part proposes the extension for the IFC-EXPRESS schema (Section 5.2.1), and then executing these extensions in the IFC-STEP file is discussed in the second part (Section 5.2.2).

5.2.1 Versioning-based IFC-EXPRESS schema

This section provides the outline implementation of the versioning concept in the IFC-EXPRESS schema through extending the schema. The first stage was to identify the elements and features information in the IFC-EXPRESS (section 5.2.1.1), then explain the concept of the proposed extension (section 5.2.1.2) and extending the IFC-EXPRESS schema to deal with the versioning concept (section 5.2.1.3).

5.2.1.1 Extract the Physical Product Information

Based on the data schema architecture of the IFC EXPRESS discussed in section 3.3.2, all building elements are within the rooted entities and derived mainly from the *IfcRoot* entity, whereas all element features within the non-rooted entities are derived from different resources layer and have completely different IFC schema. The schema of the IFC does not define a direct link between the entities that represent the building elements (e.g. *IfcBeam*) and their features (e.g. *IfcIShapeProfileDef*). In order for the IFC schema to deal with all changes in the information model (elements and features), the top entity in the IFC schema of each required resource layer are identified and extracted, which the feature entity is a child entity of that resource.

After evaluating the entities that represent the building elements and element features within the current IFC schema in chapter 3, the following items have been identified as relevant to the concept of versioning and change management:

- ***IfcBeam, IfcColumn, IfcSlab, IfcWall, IfcFooting, IfcPile, etc…*** represent shared building elements (beams, columns, slabs, and walls), and structural building elements (footing and pile). The shared building elements related to *IfcSharedBldgElements* schema (interoperability layer) while the specific building elements instructed from the *IfcStructuralElementsDomain* schema (domain layer) (Section 3.3.2.1). The entities in the both layers are within the rooted entities and both are sub-entities from *IfcBuildingElement* entity. To make selecting elements more general, *IfcElement* is used, which is a super entity of *IfcBuildingElement* and the generalization of all components that make up an AEC product. The definition of the above entities in EXPRESS ISO 10303-P11 are given below (also showed in Figure 3.3):

ENTITY *IfcBeam, IfcColumn, IfcSlab, IfcWall, IfcFooting, or IfcPile*
    ENTITY *IfcRoot*;
    GlobalId : *IfcGloballyUniqueId*;

OwnerHistory : *IfcOwnerHistory*;
Name : OPTIONAL *IfcLabel*;
Description : OPTIONAL *IfcText*;
ENTITY *IfcObjectDefinition*;
ENTITY *IfcObject*;
ObjectType : OPTIONAL *IfcLabel*;
ENTITY *IfcProduct*;
ObjectPlacement : OPTIONAL *IfcObjectPlacement*;
Representation : OPTIONAL *IfcProductRepresentation*;
ENTITY *IfcElement*;
Tag : OPTIONAL *IfcIdentifier*;
ENTITY *IfcBuildingElement*;
ENTITY *IfcBeam; IfcColumn; IfcSlab; IfcWall; IfcFooting; or IfcPile*
END_ENTITY;

- ***IfcProfileDef*** is the super entity for the definitions of the 2D geometric shapes in the profile resource. Most of the shapes (such as 'I', 'L', 'C', 'rectangle', 'circle' shaped section) in the commonly used standards are described in the sub-entities of the *IfcProfileDef*.

  Different ways are available in IFC for describing the generation of 3D solid shapes (Section 3.3.2.3.2"i"), such as "Brep", "CSG", "Swept Solid" and other solid representation. The swept solid "*IfcSweptAreaSolid*" is the preferred geometric shape representation for the building elements (Liebich, 2009, Eastman et al., 2011b). It can easily extend a series of 2D shapes into 3D geometry by sweeping one or more section profiles through a giving direction and length of the extrusion. Both longitudinal and transverse elements (such as beams and columns) and plate elements (such as walls and slabs) can be defined using the swept solid geometry. Therefore, swept solid is the shape type that is dealt with in this thesis. A standard set of commonly used section profiles required to generate the 3D shape for the building elements can be represented in *IfcProfileDef***.** *IfcIShapeProfileDef*, for example, is an entity extracted from the parent entity *IfcProfileDef* to define the parameters of all 'I' section profile. The definition of *IfcIShapeProfileDef* entity in EXPRESS ISO 10303-P11 demonstrated below to show that the *IfcProfileDef* is in the top in the hierarchy entities of the profile resource (also showed in Figure 3.5):

ENTITY *IfcIShapeProfileDef*;

  ENTITY *IfcProfileDef*;

  ProfileType : *IfcProfileTypeEnum*;

  ProfileName : OPTIONAL *IfcLabel*;

  ENTITY *IfcParameterizedProfileDef*;

  Position : *IfcAxis2Placement2D*;

  ENTITY *IfcIShapeProfileDef*;

  OverallWidth : *IfcPositiveLengthMeasure*;

  OverallDepth : *IfcPositiveLengthMeasure*;

  WebThickness : *IfcPositiveLengthMeasure*;

  FlangeThickness : *IfcPositiveLengthMeasure*;

  FilletRadius : OPTIONAL *IfcPositiveLengthMeasure*;

END_ENTITY;

- ***IfcRepresentationItem*** is the entity used to define all geometric or topological information of a product (such as line, surface, solid, etc…). Each product has its placement within the geometric representation context of the project (Section 3.3.2.3.2"i"). This means that the defined 3D shape in *IfcSolidModel* of the product has its own coordinate system using *IfcLocalPlacement* entity. *IfcPlacement* plays the role of a connector between the extruded shape and the coordinate system (as in Figure 3.5). In addition, it links with *IfcCartesianPoint* to define the geometric position of the shape. There are mainly two different kinds of elements: linear and planar elements. Two points at least are needed to define a linear element and three or more closed lines are needed to represent a planar element. As such, two Cartesian points are needed to define a straight beam or column and four Cartesian points to represent a quadrilateral slab or wall (Callister and Rethwisch, 2007, Tai and Zou, 1996). The *IfcCartesianPoint* entity, which is inherited from the super-entity *IfcRepresentationItem*, can be used several times for each building element to define the coordinates of the point locations. The specification of the *IfcCartesianPoint* entity in EXPRESS is shown below, it can be seen from the *IfcCartesianPoint* specification that the *IfcRepresentationItem* is the top entity used to define the location of all products within the geometry resource.

ENTITY *IfcCartesianPoint*;
    ENTITY *IfcRepresentationItem*;
    ENTITY *IfcGeometricRepresentationItem*;
    ENTITY *IfcPoint*;
    ENTITY *IfcCartesianPoint*;
    Coordinates : LIST [1:3] OF *IfcLengthMeasure*;
END_ENTITY;

- ***IfcMaterialDefinition*** is a general super entity for all substances that can be used to form elements in IFC (Section 3.3.2.3.2"ii"). *IfcMaterial*, For example, is the fundamental entity inherited from *IfcMaterialDefinition* used to express a single material that represents a substance for construction elements. All other entities that define the materials are related to *IfcMaterial* (as in Figure 3.7). Below is the EXPRESS specification of the *IfcMaterial* entity which shows that the *IfcMaterialDefinition* at the top in the hierarchy entities of the material resource.

ENTITY *IfcMaterial*;
    ENTITY *IfcMaterialDefinition*;
    ENTITY *IfcMaterial*;
    Name : *IfcLabel;*
    Description : OPTIONAL *IfcText;*
    Category : OPTIONAL *IfcLabel;*
END_ENTITY;

- ***IfcPropertyAbstraction*** is the top entity for the definition of some feature information (Section 3.3.2.3.2"iii"). It defines the basic concept for describing element features -based supplementary information, which specifies the set of derived measures associated with an element's physical feature (length, cost, temperature, etc.…). The feature definitions can either a single value or a list of values for specific or extended (defined by application vendors or end users) information. A sub entity *IfcPropertySingleValue* is a general entity to define a single feature as a (feature - single value) combination that provide a feature name, a description, a unit, and a nominal value (as in Figure 3.8). Below is the EXPRESS specification of the entity shows that the *IfcPropertyAbstraction* is the first entity in the definition of the property resource schema.

```
ENTITY IfcPropertySingleValue;
        ENTITY IfcPropertyAbstraction;
        ENTITY IfcProperty;
        Name                    : IfcIdentifier;
        Description             : OPTIONAL IfcText;
        ENTITY IfcSimpleProperty;
        ENTITY IfcPropertySingleValue;
        NominalValue            : OPTIONAL IfcValue;
        Unit                    : OPTIONAL IfcUnit;
END_ENTITY;
```

The same definitions of the entities above are used to redefine all the changes that are missing in the IFC file (the deleted elements and features). Table 5.1 summaries the relation between the required features by the designer in the BIM model and the entity responsible for that feature in IFC-EXPRESS schema.

Table 5.1 Element feature in BIM model and in IFC-EXPRESS schema

BIM model		IFC-EXPRESS schema		
Feature	**Example**	**Resource**	**Top Entity**	**Feature Entity**
Location	Cartesian coordinate	*Geometry Resource*	*IfcRepresentation Item*	*IfcCartesian Point*
2D Shape	I-shape/ beam	*Profile Resource*	*IfcProfileDef*	*IfcIShape ProfileDef*
Material type	Steel	*Material Resource*	*IfcMaterial Definition*	*IfcMaterial*
Others	Length, cost, colours…	*Property Resource*	*IfcProperty Abstraction*	*IfcProperty SingleValue*

5.2.1.2 *Adding the versioning concept to the IFC Standard*

The versioning concept is based around tracking and recording changes that happen within multiple models at different periods. To incorporate the versioning concept into the IFC standard, factors related to the versioning concept need to identify and combine. Some of these versioning factors are already exist in the IFC-EXPRESS schema. The available entities of the IFC EXPRESS have been used to define the rest of the versioning factors. Below are the illustrations of these factors:

1. **Version Number factor:** it represents the assigned Version Numbers of the model. Usually, a unique number is assigned to identify changes in the model. The version of the model is in the form of an incremented order (1, 2, 3,…) (Apache-Portable-Runtime, 2009). Knowing the model version makes the view or reference to the model easier later on. Two different types of model versions are represented in section (4.3.1), the global model versioning (G_v) for the inter- and intra-disciplinary designers and the local model versioning (L_v) for the intra-disciplinary designers. The increments in the numbering of both

types are separated based on the global and local changes in the information in new model.

There is no entity in the IFC-EXPRESS to define the version number of the model. *IfcLabel* in the IFC-EXPRESS is a type to represent the general information about the objects (such as name, type, role,…). This type has been used in this work to represent the version number of the model. A new number of the version (such as "version 1","version 2"…) is characterized in each type of the model versions. Therefore, this factor has a unique definition for each new model version.

2. **Ownership factor**: General information related to the current model version in terms of the creation date, creation time, disciplinary team, disciplinary designer, file name, and application name is essential to define all history and identification related project information. *IfcOwnerHistory* entity in the IFC EXPRESS with all its attributes and references entities (as presented in section 3.3.2.4) are used for the versioning purpose to identify the above information. For example, *IfcActorRole is a* reference entity *to IfcOwnerHistory* to define a set of predefined disciplines, and so on for the other information.

   Each new model version might have different ownership. The *IfcOwnerHistory* entity has been used in this work to define the general information of the current ownership for the latest model version and all previous ownerships of the previous model versions. Therefore, this factor has a unique definition for each new model version.

3. **Physical Product factor**: This has been used to identify the entities related to the shared and specific building elements as well as their features within the current IFC schema (5.2.1.1). These entities are

essential to identify the changed elements and features in the current and previous versions.

The entities that represent the building elements serve as a repository for collecting all features information (geometric shapes, spatial location, material information, and the other element features). Since the features are dealing with changeable values, then the changed values of each feature in different model versions need to be linked with each other in chronological order. Based on that, two classifications of changes are needed to be represented, (1) at the element level and (2) at the feature level. These two levels will be discussed later in more detail.

4. **Changing factor**: As declared in section (4.3.2), the changes in the elements and features can be classified into three different types that might have occurred during the last session (added, modified, and, deleted).

   IFC EXPRESS provides an enumeration type *IfcChangeActionEnum* to define the last change type that might have occurred to the elements only (but not to features). The multiple predefined changes within the enumeration type are (added, modified, deleted, no change, not defined). The first three types are further related to the changing information. These three enumeration types have been used in the versioning concept to identify the change types in the element level as well as in the feature level.

An element serves as repository to its features. Any change in the element represents change in some or all the features. To manage changes in the element level, all the features belonging to this element have to follow the action that happened to the element. This applies to the new and deleted elements (as presented in Figure 4.16). The modified element deals with some if its features. Thus, the management of information in the modified element can be done at the feature level.

As mentioned in section (4.3.2.3), the element features can be classified into permanent or temporary. The permanent feature is within the lifecycle of its element. It must be added in the new element, deleted from the removed element, or altered from the modified element. The temporary feature is optional and can be generated or removed at any time within the lifecycle of its element. It can be added in the new element, added, modified or deleted from the modified element, and deleted from the removed element. Therefore, three change options can be exhibited by the permanent feature and five options by the temporary feature. Table 5.2 summaries all options for changes in the features.

Table 5.2 Change options for the permanent and temporary features

Element	**Feature**			
	Changed			**Unchanged**
	Added	**Modified**	**Deleted**	
Added	√**(P,T)**			
Modified	√ **(T)**	√ **(P,T)**	√ **(T)**	√
Deleted			√ **(P,T)**	
P: Permanent feature T: temporary feature				

Based on the change level in the elements and features, five possible cases to link the four-versioning factors can be classified. Figure 5.1 illustrates the five cases. In the figure, only one feature is represented for each element. The red diamond represents a link to the versioning factors. The following are more elucidation of the linking cases:

- Case 1: Adding elements: this is the first step of the lifecycle for each element. All features of the added element are new information and are connected with the related element. So, the changes can be classified at the element level. The change type factor is “added” in this case and the element “A” is associated with the other versioning factors (version number, ownership, and change action type).
- Case 2: Adding features to the existing element: some features in the temporary type are new in the existing element. Therefore, the changes can be classified at the feature level. The change type factor is “added” and the feature “b_0” is associated with the other versioning factors.
- Case 3: Modifying features to the existing element: the values of some features are modified in the existing element. Consequently, the changes can be classified at the feature level. Then, the change type factor is “modified” and the value of the current feature “a_1” is associated with the other versioning factors and with the old value of the feature “a_0”.
- Case 4: Deleting features to the existing element: some temporary features are removed from the existing element. Therefore, the change can be classified at the feature level. The deleted features are outside the repository that represents the element. It is essential to be linked with its respective element. Therefore, in this case, the regenerated feature “b_0” is associated with the other versioning factors and with its element “A” to rebuild the relation between the element and the feature. The change type factor in this case is “deleted”.
- Case 5: Deleting elements: this is the last step of the element development. All features of the deleted element are removed. So, the changes can be classified at the element level. The regenerated

element (with its features) is connected with the other versioning factors and the change type factor in this case is “deleted”.

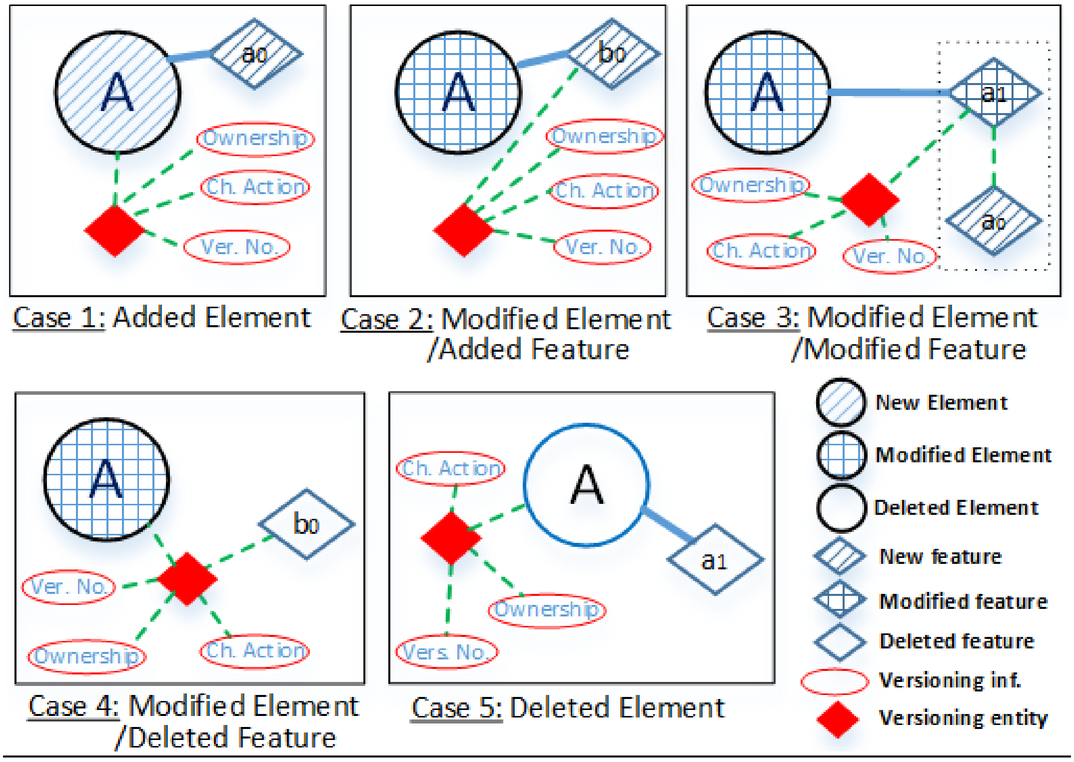


Figure 5.1 Five cases of linking the versioning factors.

5.2.1.3 PROPOSED EXTENSION FOR IFC

There are no entities in the IFC-EXPRESS schema for the purpose of managing and versioning the design changes. Based on this, an extension to the IFC EXPRESS has been suggested with a minimum set of new entities to mimic the versioning requirements. IFC4 schema is used as the basis for developing new entities –oriented data schema. A number of objectified relationships entities have been suggested to handle relationships among the different versioning factors presented above. The current objectified relationships presented in Section 3.3.2.2, which are inherited from *IfcRelationship*, do not deal with the versioning concept. Moreover, no relationship entities cater for the connection between different values of the same feature at different times.

Two relationship entities (versioning entities) have been suggested to be sub-entities of the objectified relationship entity *IfcRelationship*. This is because the information that has changed is in the element or in the feature level. The first new versioning entity that has been suggested is in the element level “***IfcRelElementChange”*** and another is in the feature level “***IfcRelFeatureChange***”. The two new versioning entities mentioned above have been collected under a new entity “***IfcRelChanges***“, which will represent the abstract generalization for collecting all the change information in IFC. *IfcRelChanges* will be the objectified relationships number seven to be inherited from *IfcRelationship*. Figure 5.2 illustrates the graphical modelling of the suggested entities.

The two suggested entities versioning “*IfcRelElementChange and IfcRelFeatureChange****”*** do not change the structure of the inheritance hierarchy of the current IFC schema (2x4). Moreover, they have nearly the same schema; the only difference is in the entities that represent the physical product factor (the elements and the features).

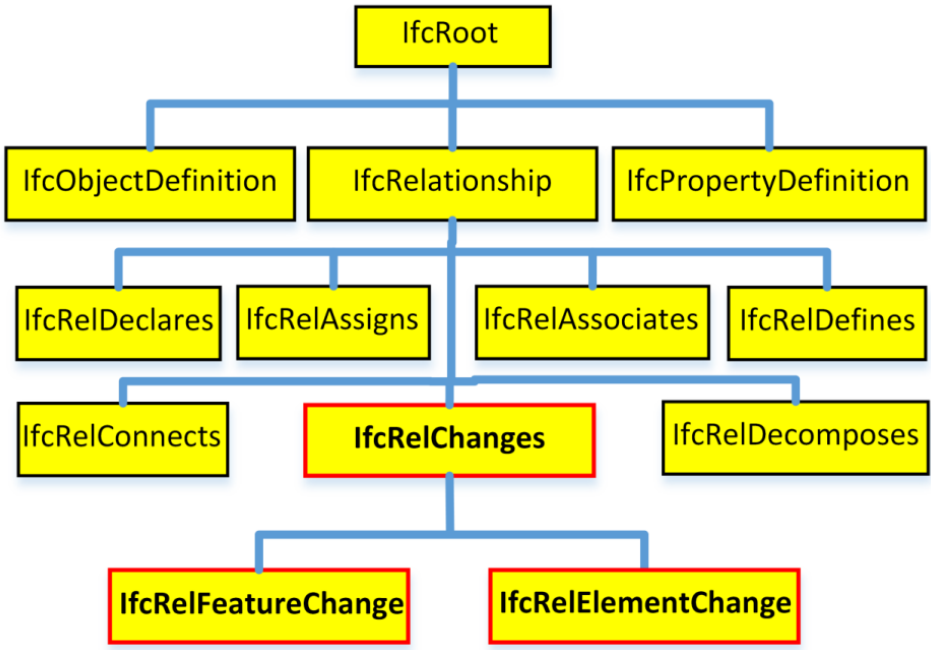


Figure 5.2: EXPRESS-G for the versioning entities

The previous sections classified five cases to link the four-versioning factors. The entities that were used for the versioning factors are *IfcLabel* for numbering the model version, *IfcOwnerHistory* for representing general information on the different model versions, *IfcChangeActionEnum* for defining the change type, and different entities for representing the building elements and the features.

The *IfcOwnerHistory* is a reference entity to the root entity *IfcRoot*. Thus, it displays in all rooted entities with the IFC EXPRERSS. Its uses have been expanded to represent the general information of the previous model versions in addition to the use for the current model. The (*IfcLabel*, and *IfcChangeActionEnum*) have been used as a direct attributes within the definition of each new versioning entity. The same for the entities that represent the physical product factor (*IfcElement, IfcProfileDef, IfcRepresentationItem, etc.…)*. They are used as reference entities within the definition of new entities. The new versioning entities are discussed in more detail in the next section. Two methods to define and represent the new EXPRESS entities have been used, textually and graphically. The textual representation of “EXPRESS” is clearer to show the inheritance hierarchy while the graphical representation “EXPRESS-G” is more suitable for explanation.

- ***IfcRelElementChange***

This is a versioning relationship entity at the element level. It is used to link the version number of the model (*IfcLabel*), the owner of the specific version (*IfcOwnerHistory*), the change action type (*IfcChangeActionEnum*), with an element or a set of elements (*IfcElement*) that has been changed. Using *IfcElement* makes the new entity more generic to cover all element information in the standard. Within the possible versioning cases (1-5) presented in the last section to link different versioning factors, case one and five, which classify changes at the element level, are applicable with this entity. Defining the entity at the element level makes the features of that element included implicitly with the changed information. The proposed definition of *IfcRelElementChange* entity in EXPRESS and EXPRESS-G (Figure 5.3) are shown below:

ENTITY *IfcRelElementChange*;
ENTITY *IfcRoot*;
GlobalId : *IfcGloballyUniqueId*;
OwnerHistory : OPTIONAL ***IfcOwnerHistory***;
Name : OPTIONAL *IfcLabel*;
Description : OPTIONAL *IfcText*;
ENTITY *IfcRelationship*;
ENTITY *IfcRelChange*;
ENTITY *IfcRelElementChange*;
VersionNumber : ***IfcLabel***
ChangeAction : ***IfcChangeActionEnum***;
ChangingElements : SET [1:n] OF ***IfcElement***
END_ENTITY;

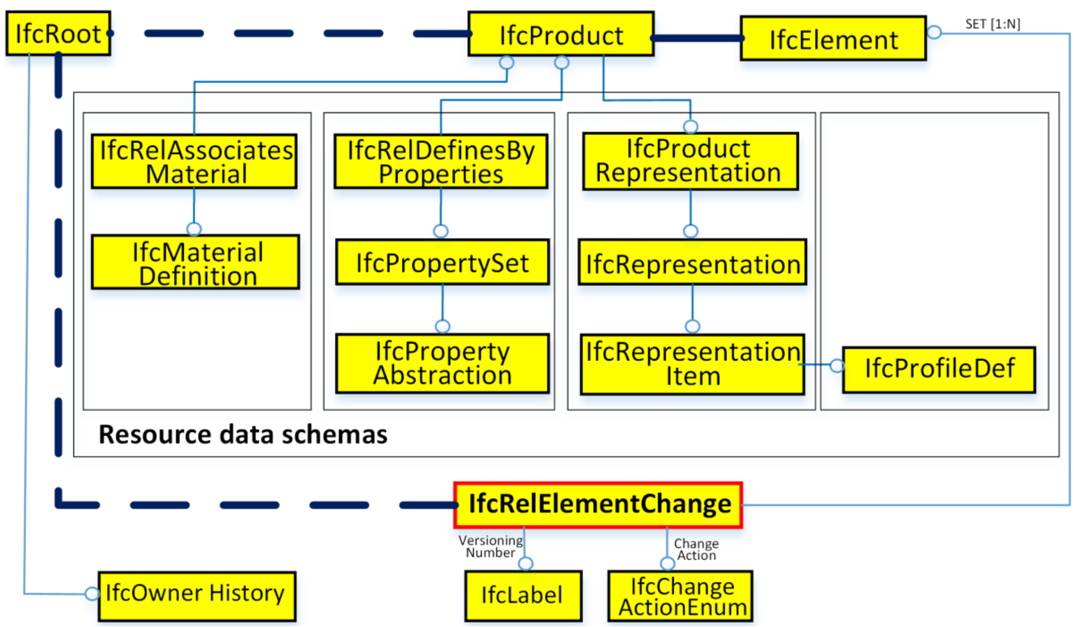


Figure 5.3: EXPRESS-G for the *IfcRelElementChange* entity.

- ***IfcRelFeatureChange***

This is a versioning relationship entity at the feature level. It links the four-versioning factors that represent the versioning concept. The physical product factor in this case is the resources entities that represent the element features (*IfcRepresentationItem, IfcPropertyAbstraction, IfcMaterialDefinition,* and *IfcProfileDef*). Choosing the top entities of these resources makes the new entity more generic to cover all feature information in the standard.

Among these four resources entities, one of them is needed to select each time to represent the required feature. There is an entity "*IfcDefinitionSelect*" with the IFC-EXPRESS that can provide the option to either select an object from the "*IfcObjectDefinition*" or a property set from the "*IfcPropertyDefinition*". The same concept is used to generate a new entity "***IfcResourceSelect***" to select one of the resources entities above. The EXPRESS Specification of *IfcResourceSelect* is illustrated below:

```
TYPE IfcResourceSelect = SELECT (
        IfcRepresentationItem,
        IfcProfileDef
        IfcMaterialDefinition
        IfcPropertyAbstraction);
END_TYPE;
```

The values of the same feature, which represent the final state with the previous states in different periods, are required to interrelate. Those previous feature versions are old information in the process of versioning, but are missing information in the current IFC file. Within the new entity "*IfcRelFeatureChange*", the values of the same feature in *IfcResourceSelect* need to be defined twice. The first value is for the old (previous) feature version and the second value is for the new (next) feature version. The new versioning entity creates a sequential and an association relationship between two values of the same element feature.

Cases two, three, and four of the versioning factors are relevant to this entity, which represent the changes at the feature level. The other versioning information that is related to the model version number, the owner and the change type belong to the second value (new) of the feature.

For case four, the relation between the entity that represent the deleted feature and the entity that represent its element within the IFC schema has been lost. The new feature entity that defines the value of the deleted feature is not within the element schema. This relation between the element and feature needs to rebuild. Since the deleted feature is defined within the versioning entity "*IfcRelFeatureChange*", the element entity "*IfcElement"* has been added to the definition of the versioning entity. As a result, this relation is rebuilt between them within the new entity. The definition and the graphical representation Figure 5.4 of the new entity are as suggested below:

```
ENTITY IfcRelFeatureChange;
    ENTITY IfcRoot;
    GlobalId            : IfcGloballyUniqueId;
    OwnerHistory        : OPTIONAL IfcOwnerHistory;
    Name                : OPTIONAL IfcLabel;
    Description         : OPTIONAL IfcText;
    ENTITY IfcRelationship;
    ENTITY IfcRelChange;
    ENTITY IfcRelFeatureChange;
    VersionNumber       : IfcLabel
    ChangeAction        : IfcChangeActionEnum;
    RelatedElement      : IfcElement
    ChangingFeatures    : OPTIONAL LIST [2] OF IfcResourceSelect
END_ENTITY;
```

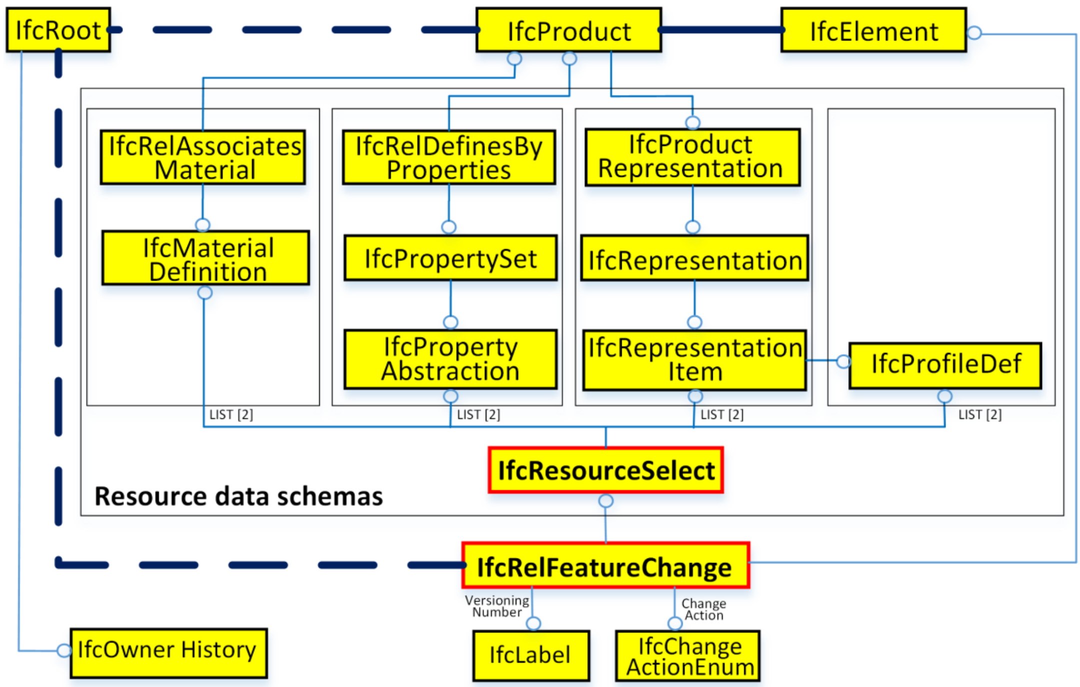


Figure 5.4: EXPRESS-G for the *IfcRelFeatureChange* entity

5.2.2 Versioning-based IFC-STEP file

The default file format to store IFC data according to the IFC-EXPRESS specification is IFC-STEP file (STEP physical file) (as illustrated in section 3.3.3). It is in text file (ASCII code) format as defined by ISO 10303-21. Therefore, the contained information can be opened in any text-based programme, such as Notepad. Each line in the file “called entity instance” typically consists of a single entity record in a compact and readable form. A program has been therefore implemented in C# to extract the required information from the file and to add the versioning information to the file. Next chapter demonstrates in detail the program implementation.

This section demonstrates how the proposed extension to the schema (the versioning entities that described above) is incorporated in the IFC-STEP file. The scenario begins with detecting the changes in information from the current model version (Section 5.2.2.1), then adding the versioning entities to the IFC-STEP file and link them with the current changes (Section 5.2.2.2) and earlier changes (Section 5.2.2.3) at the element and feature levels. Using the Evolution Graph in the developed IFC model is presented in (Section 5.2.2.4).

5.2.2.1 *Detecting the changes information*

The first step in adding the versioning concept into the STEP file is to detect and extract the changes in the entity instances that represent the versioning factors (Section 5.2.1.1 and 5.2.1.2). These changes are further related to elements and features instances, the type of change, and the general information about the current model version. Figure 5.5 illustrates the information required to be detected in the IFC-STEP file to be used in the next step.

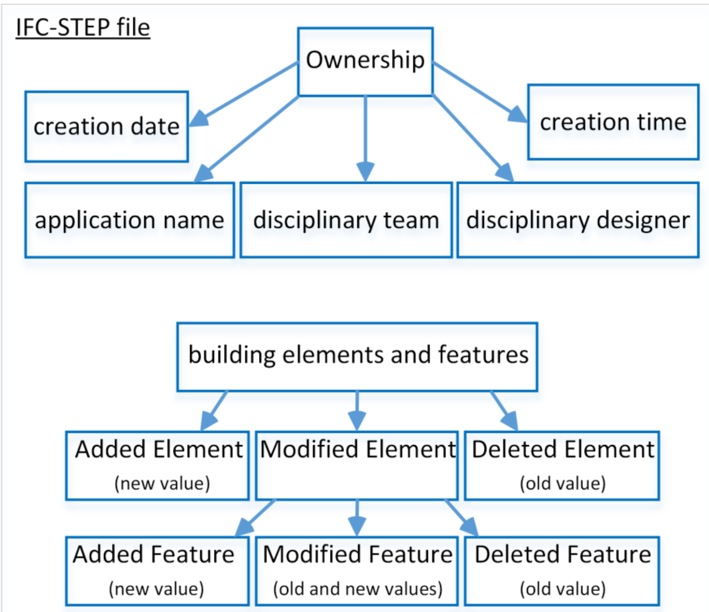


Figure 5.5: Detect the changes information in IFC-STEP file

Below is some entity instances cut from the IFC-STEP file that needs to be detected and extracted from IFC. These instances representing the ownership instance, the building element instance, and the location feature instance respectively.

Ownership

#6= IfcOwnerHistory(#5,#2,$,$,$,$,$, 201210032115);

IfcElement (element) IFC-GUID BIM-UID

#45= IfcBeam('1c_dYNPhzAtwR9KJGh3Sh1',#6,'',$,'',#42,#43, '100003');

IfcRepresentationItem (feature)

#20= IfcCartesianPoint((171.900000000002,-0.,-152.400000000005));

Instance No. Entity instance

The IFC-Standard provides a unique identification for the rooted entities, known as a globally unique identifier “GUID”. The resulting IFC-GUID is a fixed 22-character length string. The IFC-GUIDs are for exchange purposes and developed within the structure of the IFC instances. This led to providing a unique identification for each element within all IFC-STEP files. Elements in BIM model are characterized by having an own identification called unique identifier “UID”. It is 6-digits developed by the compiler of the BIM application. Both the IFC-GUID and BIM-UID for each element are recorded within the instance of that element in the IFC-STEP file. The BIM-UID for each element is different across different BIM applications while the IFC-GUID is the same. Thus, the IFC-GUIDs for the physical information are used for the analysing, identifying, and comparing purposes.

The process of detecting changes starts with analysing the current and previous IFC-STEP files separately to extract all the physical information (Section 5.2.1.1). The detect process for extracting the elements, IFC-GUIDs, and features information from IFC model can be classified as steps:

- Step 1: Find entity instance that represents an element (e.g. *IfcBeam*, *IfcColumn*, etc.….).
- Step 2: Find IFC-GUID of that element in the same element instance.
- Step 3: Find entity instances that represent the element features (*IfcCartesianPoint*, *IfcIShapeProfileDef* *IfcPropertySingleValue*, and

IfcMaterial). The relations between the building elements with their features within the IFC schema are used to find the element features (Section 3.3.2.3.2).

- Step 4: the three previous steps are repeated to find the other elements and features.

A part of IFC-STEP file is illustrated in Figure 5.6 as an example to find an element entity "*IfcBeam*", its IFC-GUID and all its features.

```
③ #280= IFCMATERIAL('Metal - Steel 43-275');
  #281= IFCCOLOURRGB($,0.,0.,0.);
  #282= IFCDRAUGHTINGPREDEFINEDCURVEFONT('continuous');
  ...............................
  #692= IFCCARTESIANPOINT((3778.33952514916,5277.54788857509,4000.));
  #694= IFCAXIS2PLACEMENT3D(#692,$,$);
  #695= IFCLOCALPLACEMENT(#59,#694);
  #696= IFCCARTESIANPOINT((-5.40012479177676E-13,-2.70006239588838E-13));
  #698= IFCAXIS2PLACEMENT2D(#696,#19);
③ #699= IFCISHAPEPROFILEDEF(.AREA.,'UB305x165x54',#698,166.8,310.9,7.7,13.7,8.9);
  #700= IFCCARTESIANPOINT((172.,-0.,155.457));
  #702= IFCAXIS2PLACEMENT3D(#700,#9,#13);
  #703= IFCEXTRUDEDAREASOLID(#699,#702,#17,3656.);
  #704= IFCSTYLEDITEM(#703,(#193),$);
  #706= IFCSHAPEREPRESENTATION(#44,'Body','SweptSolid',(#703));
③ #708= IFCCARTESIANPOINT((3778.33952514914,5277.5478885751));
  #710= IFCCARTESIANPOINT((7778.33952514914,5277.5478885751));
  #712= IFCPOLYLINE((#708,#710));
  #714= IFCSHAPEREPRESENTATION(#43,'Axis','Curve2D',(#712));
  #716= IFCPRODUCTDEFINITIONSHAPE($,$,(#714,#706));                    ②
① #718= IFCBEAM('1pSs3fD0n5gw6IhATbSOx7',#52,'UB-Universal
  Beams:UB305x165x54',$,'UB-Universal Beams:UB305x165x54:227884',#695,#716,'212478');
  #719= IFCPROPERTYSINGLEVALUE('End Extension',$,IFCLENGTHMEASURE(-12.7),$);
③ #720= IFCPROPERTYSINGLEVALUE('Work Plane',$,IFCLABEL('Level : Level 1'),$);
  #721= IFCPROPERTYSINGLEVALUE('Cut Length',$,IFCLENGTHMEASURE(3656.),$);
  #722= IFCPROPERTYSINGLEVALUE('Volume',$,IFCVOLUMEMEASURE(24689955.1200001),$);
  #723= IFCPROPERTYSINGLEVALUE('Start Extension',$,IFCLENGTHMEASURE(-12.7),$);
  #724= IFCPROPERTYSET('38zZwrixvFAhnF2iWBB$OL',#52,'Other',$,(#721,#722));
  #726= IFCRELDEFINESBYPROPERTIES('2Eg2yD6Zr30BMWi8TGbzvl',#52,$,$,(#718),#724);
  ..................................................
  #4033=
  IFCRELASSOCIATESMATERIAL('1Iq0rRvYTBa9ETjWmoUgXC',#52,$,$,(#.......,#718,#.....),#280);
```

Figure 5.6 Extract element, IFC-GUID, and feature information from IFC file

The process of comparing and matching the current with the predecessor model versions to detect the changes will be discussed in the next chapter.

5.2.2.2 *Adding the versioning factors*

After evaluating the IFC files and identifying the change entity instances (elements and features), the following step is to integrate the versioning concept into the existing IFC model.

Beside the original information in the proposed extension to the IFC-STEP file, the versioning information would have a section added to the file. The IFC-STEP file has been divided into two main parts: **original and extended parts**. The original part represents the existing entity instances in the IFC file that can be read by the compatible software applications. The contents of the IFC in this part represent the current state of the model information. The extended part represents managing the changes information that have been found in the current model version (the original part) and all the previous changes in the earlier model versions. The extended part is nested with the original part to form together the current and the change information. Dividing the STEP file into two parts was only for illustration purposes. Figure 5.7 illustrates a set of model versions (M_0 to M_3) to clarify the original and the extended parts in IFC.

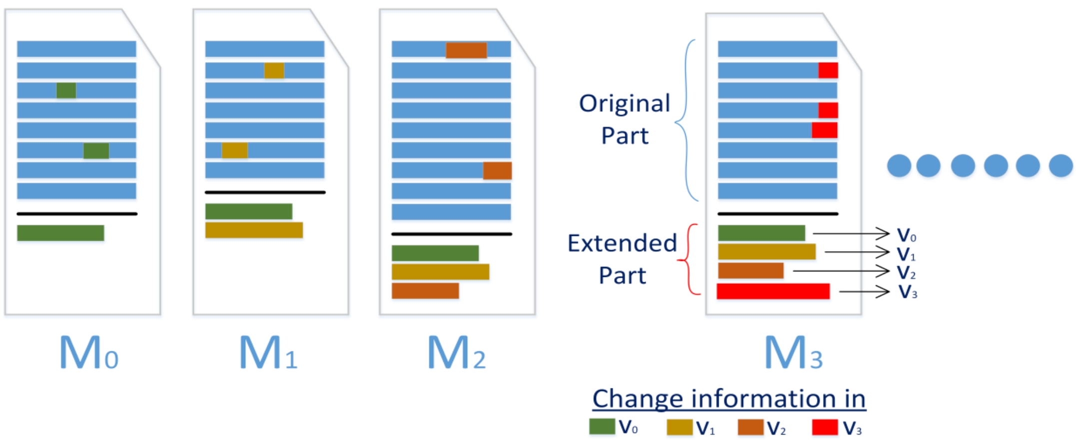
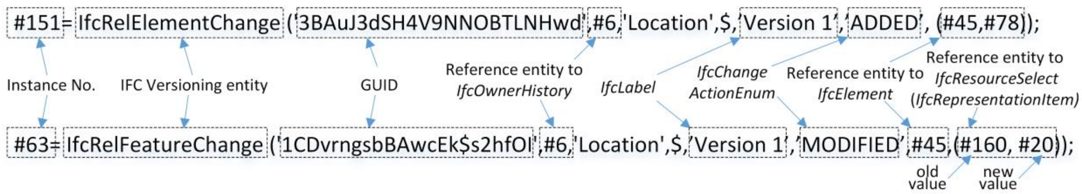


Figure 5.7: Graphical representation of original and extended parts in IFC.

The current changes in the entity instances (not the deleted information) are available information in the original part of the IFC file, which need to be detected from the file. In contrast, the deleted and previous changes are missing information, which needs to be generated in the extended part of the IFC file.

Based on the changed information that was collected, several versioning entities at the element level "*IfcRelElementChange*" and at the feature level "*IfcRelFeatureChange*" are established and used to make up the extended part in the current model version. These versioning entities are used to gather the versioning factors (*IfcLabel* to number the model version, *IfcChangeActionEnum* to define the change type, *IfcOwnerHistory* to represent the general information, and *IfcElement* for changed elements and/or *IfcResourceSelect* for changed features). Below are the proposed instances that represent the versioning entities culled from the IFC file.



Several flow charts are presented below to illustrate the implementation aspects of this proposal. This type of diagram was adopted to achieve a good degree of simplicity and clarity in presentation. The changes identification and the versioning generation flowchart are given in Figure 5.8 to achieve the versioning concept in the IFC-STEP file. The flowchart consists of comments along with the original part of the IFC file to capture the general information about the current model version and define a version number for the model.

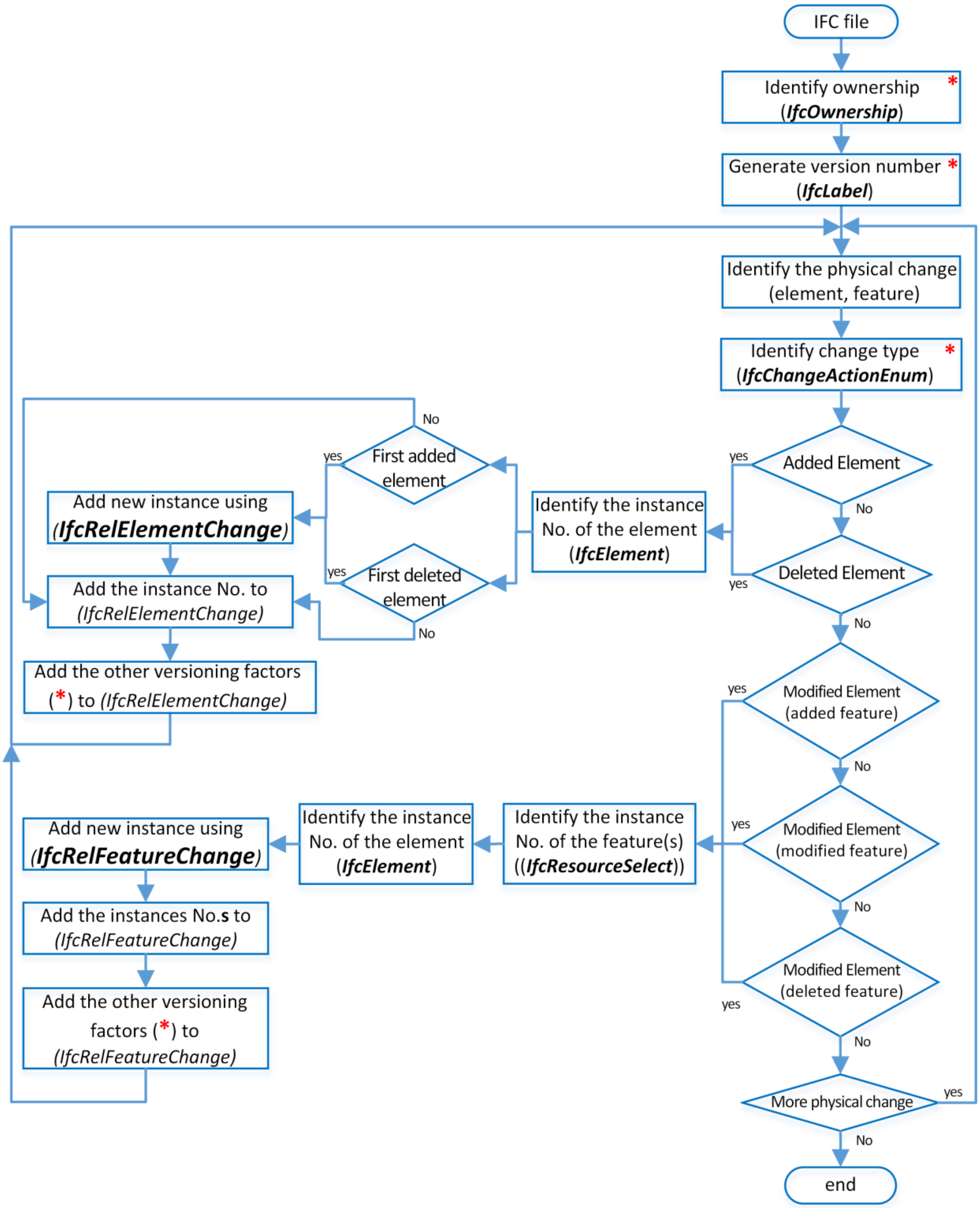


Figure 5.8: change identification and versioning generation flowchart

The next step in the sequence of events is to detect the changes in the elements and features. To handle each single change separately in the model, the level of change in the element or in the feature must be specified. Based on the change level, the two-versioning entities are used. The added and deleted elements are changes at the element level. Therefore, a unique entity instance for the version entity

"*IfcRelElementChange*" is generated in the current model version to collect all added elements and link them with the other versioning factors. If a single element or more is deleted, the same version entity is generated for the deleted elements. Accordingly, two instances of the "*IfcRelElementChange*" version entities have been generated in each new model version.

On the other hand, since the modified element is not dealing with all its features, the change type is placed at the feature level. Therefore, for each change at the feature level, a new version entity "*IfcRelFeatureChange*" will be generated to collect the two values of the same feature in an ordered way and link them with the related element and with the other versioning factors.

The five cases presented in section 5.2.1.2 are used to illustrate the change level for the element and feature. In the previous flowchart (Figure 5.8), the information between: identifying the changing type of the element or feature, and identifying the instance number of the element (*IfcElement*) has been expanded. Therefore, the flowcharts below represent the required information based on the physical product only excluding the other versioning factors (version number, ownership, and changing type).

- **Added element**: the entity instance for the added element and all its features are available in the original part of the IFC file. Based on the data schema architecture of the IFC EXPRESS, The features are connected through the entity references to the related element in the original part. Therefore, the only change information that is needed for the *IfcRelElementChange* is the instance number of the new element. The same versioning entity is used to record the other instance numbers of the new elements.

- **Deleted element**: the entity instance for the deleted element and all its features are missing information in the current IFC-STEP file. Therefore, these instances are needed to be regenerated and recorded in the extended part. The existing definition in the IFC EXPRESS schema (Section 3.3.2.1) is used to regenerate the deleted information. The instance number of the deleted element is used in the *IfcRelElementChange*. The same versioning entity is used to record the other instance numbers of the deleted elements of the same model version. Figure 5.9 illustrates the *IfcRelElementChange* flowchart.

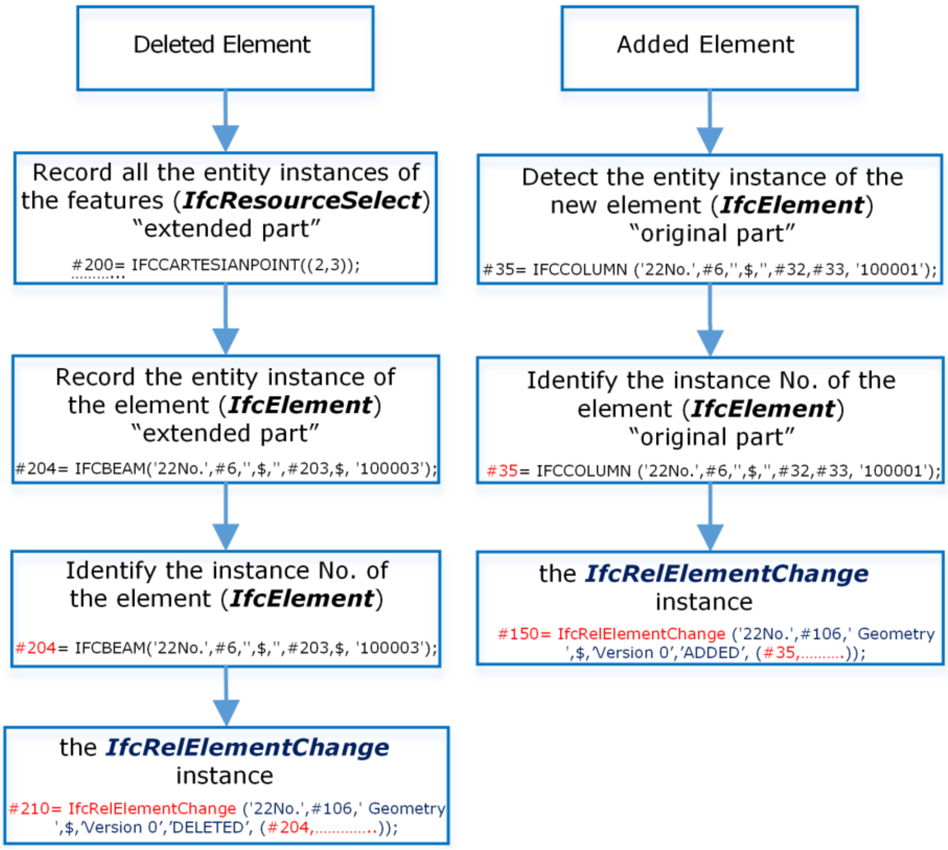


Figure 5.9: *IfcRelElementChange* flowchart

- **Modified element/ added feature**: the entity instance that represents the added feature of the existing element is available in the original part of the IFC file. Based on the proposed data schema for the *IfcRelFeatureChange*, two values of the same feature are required. The old feature value in the previous model version and the new feature value in the current model version. IFC EXPRESS schema provides the possibility of not filling the selected attributes within the definition of the entity. Therefore, the "optional attribute" has been used with the "Changing Features" in the definition of the new version entity "*IfcRelFeatureChange*" (Section 5.2.1.3). Logically, there is no old value for a new feature. The "optional" advantage in IFC schema is used to denote the old value. "$" character is used in IFC-STEP file to encode the indefinite values. As a result, the only required information for the *IfcRelFeatureChange* instance is the instance number of the added feature as well as the instance number of the element.

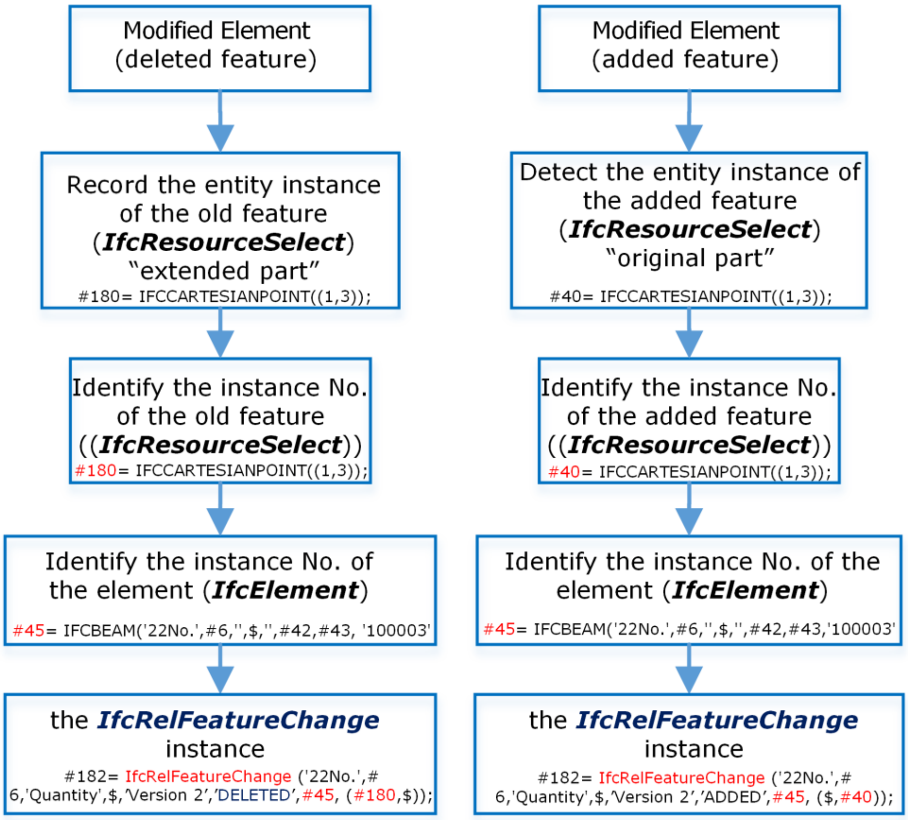


Figure 5.10: *IfcRelFeatureChange* flowchart (added and deleted features)

- **Modified element/ deleted feature**: the entity instance for the deleted feature is missing information in the current IFC-STEP file. This instance is needed to be documented in the extended part of the current IFC file.

Since there is no new value for a deleted feature in the current model version, the optional advantage in IFC schema is used to denote the new value. The instance numbers of the deleted feature besides its element are required in the *IfcRelElementChange* instance. Figure 5.10 illustrates the *IfcRelFeatureChange* generation flowchart for the added and deleted features.

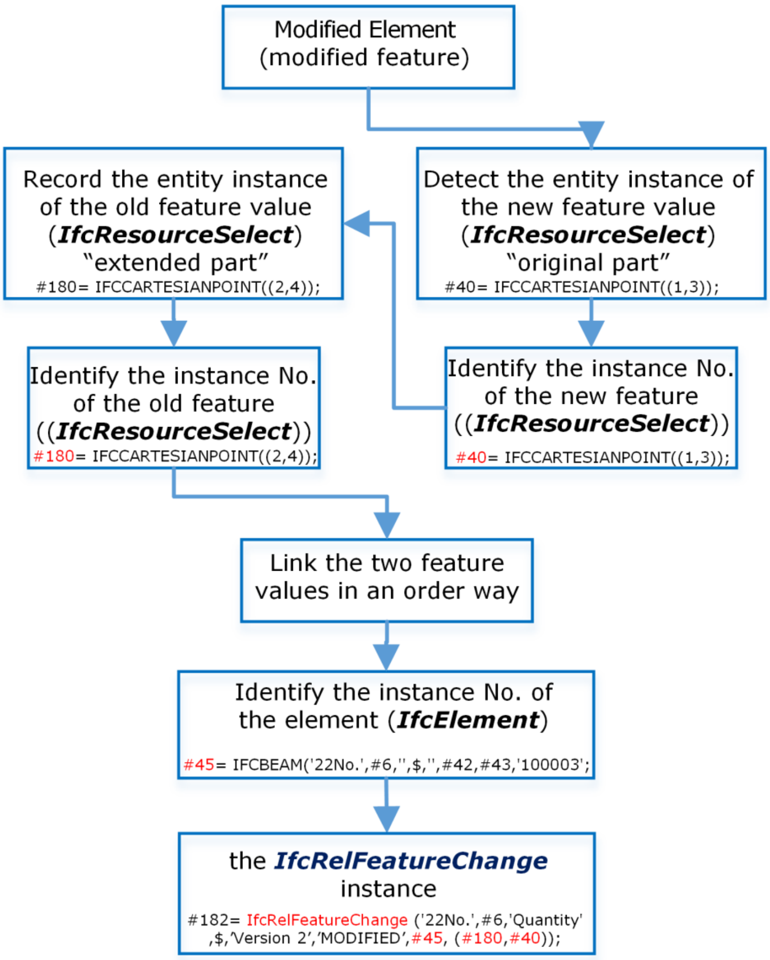


Figure 5.11: *IfcRelFeatureChange* flowchart for modified features

- **Modified element/ modified feature**: the modified feature is dealing with two values (old and new). The entity instance of the new feature value is available in the original part of the IFC file while the entity instance of the old feature value is missing information in the current IFC-STEP file. The missing instance needed to be regenerated and recorded in the extended part of the current IFC file. The instance numbers of the new and deleted feature values plus their element are added to the *IfcRelElementChange* instance. Figure 5.11 illustrates the *IfcRelFeatureChange* generation flowchart for the modified features.

5.2.2.3 Linking the versioning information

Any engineering project addresses sets of sequential versions of the model issued during different periods by multi-disciplinary teams. The last section demonstrates versioning the current changes in the model version. This section demonstrates dealing with the old changes in all previous model versions and linking them with the current changes in the latest model version.

In each model version, the extended part is generated to gather all changes in that model version based on the comparison with the previous version. A series of changes might have occurred to an element or a specific feature during its life. These changes might be available in different (not continuous) model versions. The extended part in the current IFC-STEP file can be more expanded to cover the earlier changes of the model versions at the element and feature levels. Therefore, all earlier elements and features versions need to be linked together in the IFC-STEP file and regularly updated their instance numbers in each new version of the model to record the new design changes.

The instances of the two-versioning entities (*IfcRelElementChange* and *IfcRelFeatureChange*) can be used many times in the "extended part" of the IFC file. The new value of the feature in *IfcRelFeatureChange* instance for the model version will be an old value in another *IfcRelFeatureChange* instance for the next feature version. Therefore, through sharing the same feature value (through the instance number) in two *IfcRelFeatureChange* instances and through defining the same element in different *IfcRelElementChange* and *IfcRelFeatureChange*, an interrelated chain for the change information can be built. Different changes are proposed in a simplified example (Figure 5.12) for a model “M” with a single element “E” and two features information “a and b” to present the new versioning entities, the five versioning cases, and to show the links between features. “a” is permanent feature and “b” is temporary feature.

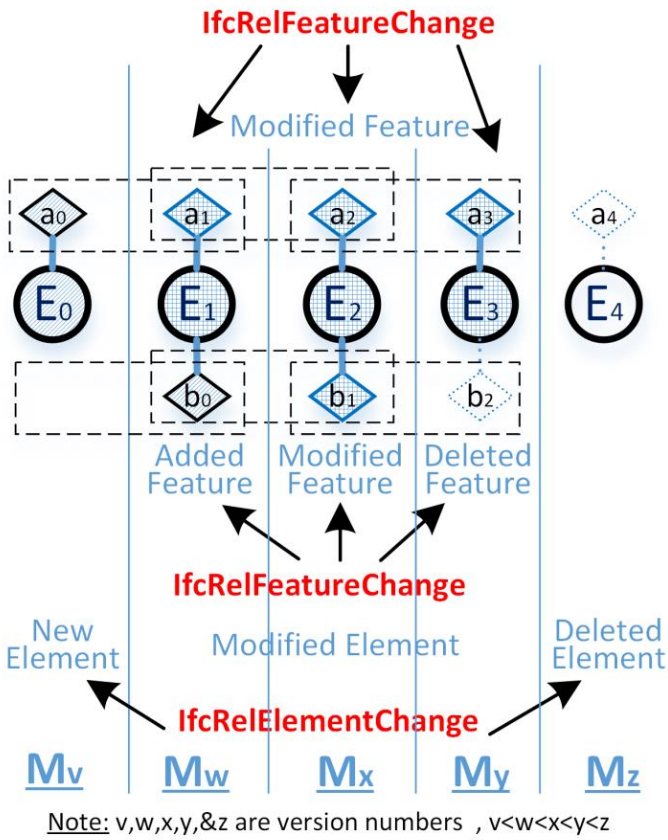


Figure 5.12 Linking the versioning information

The “extended part” of the previous model version can be attached and linked with the next model version to produce the new “extended part” in the current model version. Because of the cumulated changes, the amount of information in this part will keep increasing with the progress of the design. Usually, in any design process, the ratio between the design changes and the whole design model is very small. On the other hand, the new versioning entities have been designed to use minimum amount of information to be sufficient to record the changes. Therefore, the extended part, even its information is cumulative, compared with the original part is remained small.

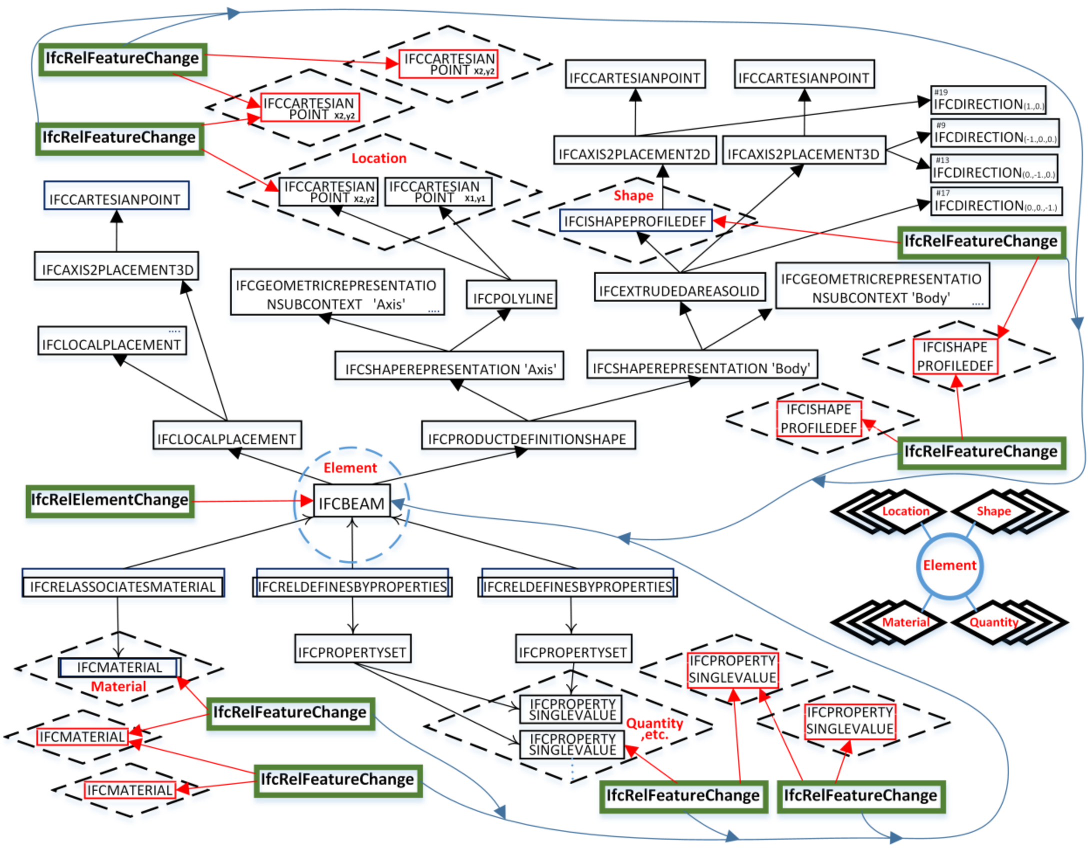


Figure 5.13: Graphical modelling of a beam in the developed IFC-STEP file

Figure 5.13 illustrates the graphical relation of the IFC entities in part of the developed IFC STEP file to define a specific building element, which is represented by "*IfcBeam*". The entities that represent the features are not directly connected to the entity that represents the specific element. Each feature has a map of a complicated combination of entities to represent its feature. Information about engineering features (such as, location, cross-section, material, and quantity) to the building elements is documented within the EXPRESS definition. The black boxes represent the entities in the original part, while the green and red boxes are in the extended part. The red boxes are the old feature versions and the green boxes are the versioning entities. The red arrows represent linking two values of the feature instances for the same feature and the blue arrows represent linking the features values with their element (*IfcBeam*).

5.2.2.4 Evolution Graph and the developed IFC model

The evolution graph of section 4.3.2.4 can be used with the developed IFC model to show the relation between the available information in the graph and in the "original and extended part" of the IFC model. Any model version in the Model Evolution Graph represents the changed and unchanged information of the current state for the elements and features. To this end, the current information in the "original part" of the IFC model represents the last model version in the model evolution graph that the element and feature versions are unknown.

On the other side, any model version in the Change Evolution Graph represents the changed information (including the deleted information) for the elements and features. In that case, the information in the "extended part" of the IFC model represents the whole model versions in the Change Evolution Graph. To illustrate this, the element G in Figure 4.17 and 4.18 is taken as an example. When M_3 is the latest model version, element G in M_3 (Figure 4.17) equals to the G information in the "original part" of the IFC model (but without the versioning numbers of the element and the features) and element G from M_{0C} to M_{3C} (Figure 4.18) equals to the G information in the "extended part" of the IFC model.

Figure 5.14 shows the representation of the developed IFC model for the element G in the evolution graph. The two values of the same feature in different periods have been connected with dotted line and the value of the same feature in the model and change evolution graph have been connected with full arrow.

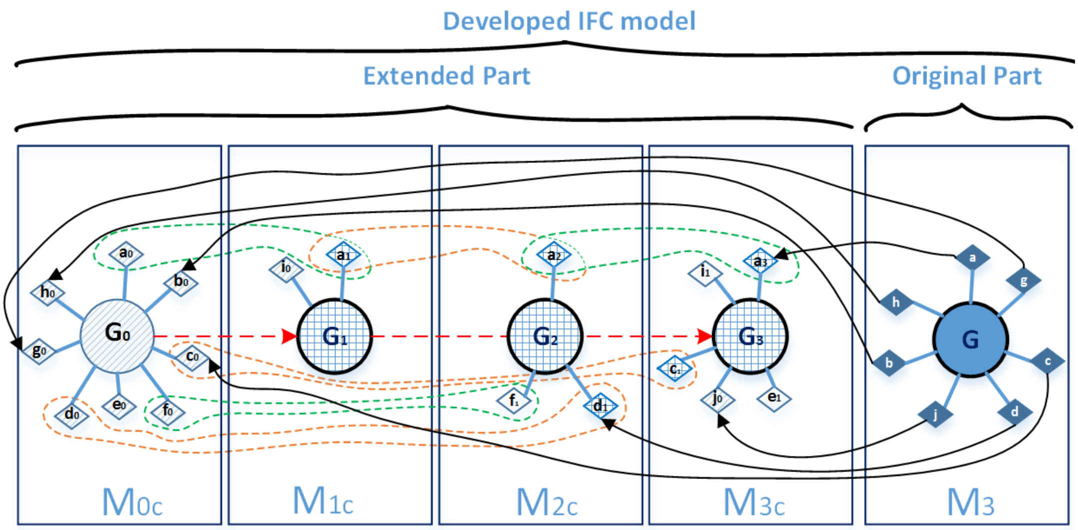


Figure 5.14: Relationship between the evolution graphs and the developed IFC model

5.3 Example

The Implementation of the versioning concept in the IFC-STEP file and the relation between the versioning instances are discussed in a simple example.

5.3.1 The goal of the example

An example involves an up-close, in-depth, and comprehensive examination (the study) of a special subject (the case) (Gillham, 2000, Scapens, 2004). Researchers appear to be in consensus that the example is an experimental method to validate the researcher aims (Easterbrook and Callahan, 1998, Stake, 2013). examples have been identified as one of the contributing methods to problem solving in requirement engineering (Olofsson et al., 2008, Barlish and Sullivan, 2012).

5.3.2 Implementing the example

To implement the versioning concept in the IFC-STEP file, a simple example is presented in Figure 5.15. Three elements (A, B, and D), which represent two columns and one beam, with two features (c, and a) for each element are proposed. The first feature (c) is a permanent feature to represent a Cartesian point and the second feature (a) is a temporary feature to represent the cross-sectional area. Four sequential model versions have been proposed (M_0, M_1, M_2, and M_3). The architect generates (M_0 and M_2) and the structural designer generates (M_1 and M_3). A set of changes have been proposed for each model version as presented in the figure.

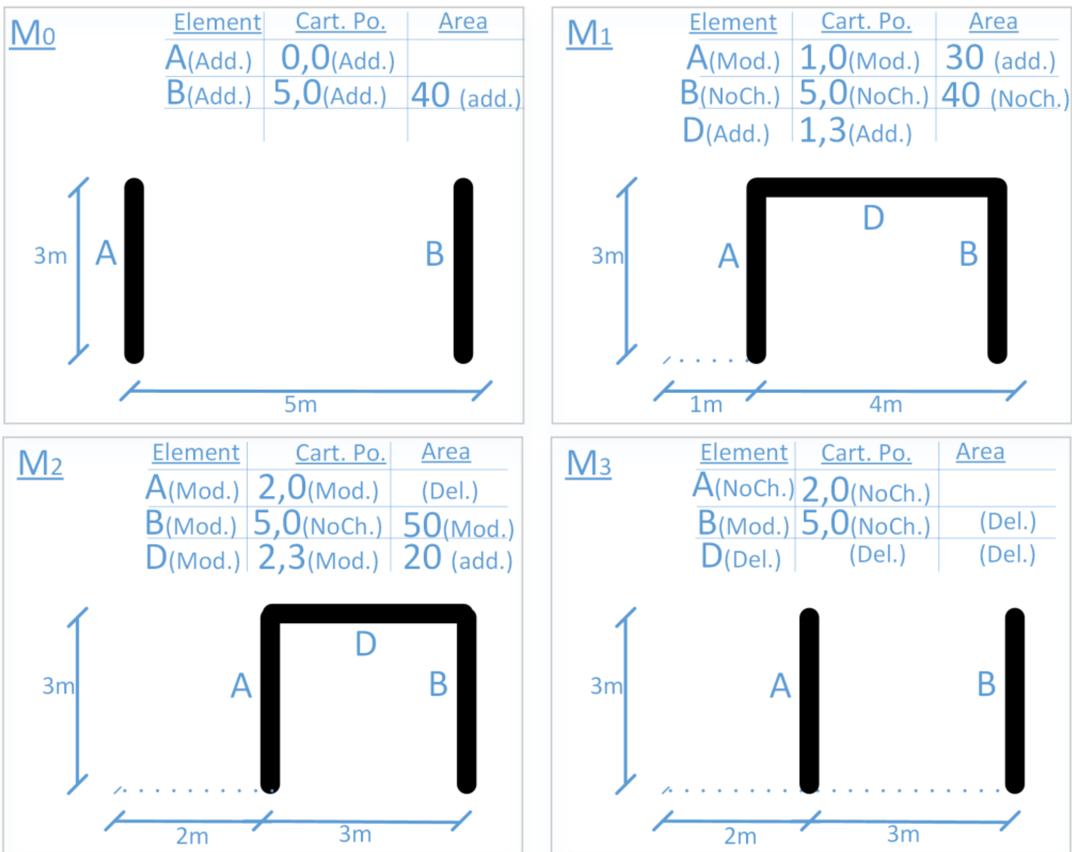


Figure 5.15: Example based on four model versions

To analyse the original and the extended parts on the IFC-STEP file, An IFC-STEP file for each model version of Figure 5.15 is prepared in Figure 5.16 and Figure 5.17. Each IFC-STEP file includes the current information of the model with the current changes and all earlier changes of the previous models. Below are some assumptions to clarify the explanation of the file contents:

- Due to the large amount of information contained in the IFC-STEP file, only the required entity instances in the original part are represented.
- The instance numbers for the entities in the IFC file is the same for the four model versions.
- To save more spaces, the 22 numbers for each GUID was changed to a fixed text “22No.” for all rooted entities.
- Asterisks “*******” were used to break the original part (the upper part) from the extended part (the lower part).
- In the extended part of the IFC-STEP file, the red font represents the current changes compared with the previous model version and the black font represent the changes in all earlier model versions.

The process of versioning of the models is illustrated below to clarify further Figure 5.15. Each of the five-versioning cases in Section 5.2.1.2 is discussed below:

1. In the prototype model (M_0),
   - All the information is new. Consequently, a single “*IfcRelElementChange*” instance is required to collect the entire new elements (A and B). With this instance, the numbering of the model version “*IfcLabel* “ is “0”, the changing type of the instance “*IfcChangeActionEnum*” is “added” and the ownership of the current change reference to the “*IfcOwnerHistory*” entity (#6).
2. In the model version (M_1),
   - The old instances that represent the ownership of M_0 are documented in the extended part of M_1 (#101 to #106).
   - A new “*IfcRelElementChange*” is used in (#151) for new element (D).

- The entity instance (#160) represents the old value of the modified feature (c) for element (A) (#20 in M_0).
- Two instance numbers (#160 and #20) are used in the *IfcRelFeatureChange* instance (#163) to represent the old and new values of the modified feature (c) of element (A). The entity instance of (#160) is generated in the extended part while the entity instance of (#20) is existed in the original part.
- In *IfcRelFeatureChange* instance (#163), The model version number 'version 1', the ownership and the change type 'MODIFIED' belong to the new value of the feature (#20).

3. In the model version (M_2),

- The ownership of this model is the same as M_0. Therefore, only the *IfcOwnerHistory* is used (#106) to record the creation date. The other information can be found it in the original part of the file (#1 to #5).
- The entity instances (#160 and #161) are the old values, and (#20) is the new value of the modified feature (c) of element (A). (#161) works as a link between (#160 and #20). It represents the new feature value in the old versioning instance (#163) and the old feature value in the new versioning instance (#164).
- The entity instance (#165), which represents the deleted feature (a) of element (A), has been recorded in the extended part to be used in the versioning instance (#168).
- "$" character is used in (#168) to encode the non-definite values, which represent the new value of the deleted feature (a) for element (A). Only the first (old) value (#165) is defined in the versioning instance (#168).
- "$" character is used in (#182) to represents the old value of the new feature (a) of element (D). Therefore, the second (new) value (#41) in the definition of the feature is the only feature information required in the versioning instance (#182).

M0

```
#1= IFCPERSON($,'MIKE',$,$,$,$,$,,$);
#2= IFCORGANIZATION($,'TIC',$,$,$);
#3= IFCACTORROLE(.ARCHITECT.,$,$)
#4= IFCAPPLICATION(#2,'2013','Autodesk Revit 2013','Revit');
#5= IFCPERSONANDORGANIZATION(#1,#2,(#3));
#6= IFCOWNERHISTORY(#5,#2,$,$,$,$,$, 201210022108);
#20= IFCCARTESIANPOINT((0,0));
#25= IFCCOLUMN('22No.',#6,'',$,'',#22,#23, '100001');            // A
#30= IFCCARTESIANPOINT((5,0));
#31= IFCPROPERTYSINGLEVALUE('Area',$, IFCAREAMEASURE (40.),$);
#35= IFCCOLUMN('22No.',#6,'',$,'',#32,#33, '100002') );          // B
*************
#150= IfcRelElementChange ('22No.',#6,' Geometry ',$,'Version 0','ADDED', (#25,#35));
```

M1

```
#1= IFCPERSON($,'ARAS',$,$,$,$,$,,$);
#2= IFCORGANIZATION($,'ABC',$,$,$);
#3= IFCACTORROLE(.STRUCTURE.,$,$)
#4= IFCAPPLICATION(#2,'2013','Autodesk Revit 2013','Revit');
#6= IFCOWNERHISTORY(#5,#2,$,$,$,$,$, 201210032115);
#20= IFCCARTESIANPOINT((1,0));
#21= IFCPROPERTYSINGLEVALUE('Area',$, IFCAREAMEASURE (30.),$);
#25= IFCCOLUMN('22No.',#6,'',$,'',#22,#23, '100001');       // A
#30= IFCCARTESIANPOINT((5,0));
#31= IFCPROPERTYSINGLEVALUE('Area',$, IFCAREAMEASURE (40.),$);
#35= IFCCOLUMN('22No.',#6,'',$,'',#32,#33, '100002');       // B
#40= IFCCARTESIANPOINT((1,3));
#45= IFCBEAM('22No.',#6,'',$,'',#42,#43, '100003');         // D
*************
#101= IFCPERSON($,'MIKE',$,$,$,$,$,,$);
#102= IFCORGANIZATION($,'TIC',$,$,$);
#103= IFCACTORROLE(.ARCHITECT.,$,$)
#104= IFCAPPLICATION(#112,'2013','Autodesk Revit 2013','Revit');
#105= IFCPERSONANDORGANIZATION(#111,#112,(#113));
#106= IFCOWNERHISTORY(#105,#102,$,$,$,$,$, 201210022108);
#150= IfcRelElementChange ('22No.',#106,'Geometry ',$,'Version 0','ADDED', (#25,#35));
#151= IfcRelElementChange ('22No.',#6,'Geometry ',$,'Version 1','ADDED', (#45));
#160= IFCCARTESIANPOINT((0,0));
#163= IfcRelFeatureChange('22No.',#6,'Geometry',$,'Version 1','MODIFIED', #25,(#160, #20));
#167= IfcRelFeatureChange('22No.',#6,' Quantity ',$,'Version 1','ADDED',#25,($,#21));// A
```

M2

```
#1= IFCPERSON($,'MIKE',$,$,$,$,$,,$);
#2= IFCORGANIZATION($,'TIC',$,$,$);
#3= IFCACTORROLE(.ARCHITECT.,$,$)
#4= IFCAPPLICATION(#2,'2013','Autodesk Revit 2013','Revit');
#5= IFCPERSONANDORGANIZATION(#1,#2,(#3));
#6= IFCOWNERHISTORY(#5,#2,$,$,$,$,$, 201210045158);
#20= IFCCARTESIANPOINT((2,0));
#25= IFCCOLUMN('22No.',#6,'',$,'',#22,#23, '100001');                  // A
#30= IFCCARTESIANPOINT((5,0));
#31= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(50.),$);
#35= IFCCOLUMN('22No.',#6,'',$,'',#22,#23, '100002');                  // B
#40= IFCCARTESIANPOINT((2,3));
#41= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(20.),$);
#45= IFCBEAM('22No.',#6,'',$,'',#42,#43, '100003');                    // D
*************
#106= IFCOWNERHISTORY(#5,#2,$,$,$,$,$, 201210022108);
#111= IFCPERSON($,'ARAS',$,$,$,$,$,,$);
#112= IFCORGANIZATION($,'ABC',$,$,$);
#113= IFCACTORROLE(.STRUCTURE.,$,$)
#114= IFCAPPLICATION(#112,'2013','Autodesk Revit 2013','Revit');
#115= IFCPERSONANDORGANIZATION(#111,#112,(#113));
#116= IFCOWNERHISTORY(#115,#112,$,$,$,$,$, 201210032115);
#150= IfcRelElementChange ('22No.',#106,' Geometry ',$,'Version 0','ADDED', (#25,#35));
#151= IfcRelElementChange ('22No.',#116,' Geometry ',$,'Version 1','ADDED', (#45));
#160= IFCCARTESIANPOINT((0,0));
#161= IFCCARTESIANPOINT((1,0));
#163= IfcRelFeatureChange('22No.',#116,' Geometry ',$,'Version 1','MODIFIED', #25,(#160,#161));
#164= IfcRelFeatureChange('22No.',#6,' Geometry',$,'Version 2','MODIFIED',#25,(#161,#20));
#165= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(30.),$);
#167= IfcRelFeatureChange ('22No.',#116,' Quantity ',$,'Version 1','ADDED', #25,($,#165));
#168= IfcRelFeatureChange('22No.',#6,'Quantity',$,'Version 2',DELETED, #25,(#165 , $);   // A
#170= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(40.),$);
#175= IfcRelFeatureChange('22No.',#6,'Geometry',$,'Version 2','MODIFIED', #35,(#170,#31));// B
#180= IFCCARTESIANPOINT((1,3));
#181= IfcRelFeatureChange('22No.',#6,'Geometry',$,'Version 2','MODIFIED', ,#45,(#180,#40));
#182= IfcRelFeatureChange('22No.',# 6,'Quantity',$,'Version 2','ADDED',#45,($, #41));  // D
```

Figure 5.16 IFC-STEP files for model versions (M_0, M_1, M_2).

M_3

```
#1= IFCPERSON($,'ARAS',$,$,$,$,$,,$);
#2= IFCORGANIZATION($,'ABC',$,$,$);
#3= IFCACTORROLE(.STRUCTURE.,$,$)
#4= IFCAPPLICATION(#2,'2013','Autodesk Revit 2013','Revit');
#5= IFCPERSONANDORGANIZATION(#1,#2,(#3));
#6= IFCOWNERHISTORY(#5,#2,$,$,$,$,$, 201210058578);
#20= IFCCARTESIANPOINT((2,0));
#25= IFCCOLUMN('22No.',#6,'',$,'',#22,#23, '100001');    A
#30= IFCCARTESIANPOINT((5,0));
#35= IFCCOLUMN('22No.',#6,'',$,'',#32,#33, '100002');    B
*************
#101= IFCPERSON($,'MIKE',$,$,$,$,$,,$);
#102= IFCORGANIZATION($,'TIC',$,$,$);
#103= IFCACTORROLE(.ARCHITECT.,$,$)
#104= IFCAPPLICATION(#2,'2013','Autodesk Revit 2013','Revit');
#105= IFCPERSONANDORGANIZATION(#1,#2,(#3));
#106= IFCOWNERHISTORY(#105,#102,$,$,$,$,$, 201210022108);
#116= IFCOWNERHISTORY(#5,#2,$,$,$,$,$, 201210032115);
#126= IFCOWNERHISTORY(#105,#102,$,$,$,$,$, 201210045158);
#150= IfcRelElementChange ('22No.',#106,' Geometry ',$,'Version 0','ADDED', (#25,#35));
#151= IfcRelElementChange ('22No.',#116,' Geometry ',$,'Version 1','ADDED', (#204));
#160= IFCCARTESIANPOINT((0,0));
#161= IFCCARTESIANPOINT((1,0));
#163= IfcRelFeatureChange ('22No.',#116,'Geometry',$,'Version 1','MODIFIED', #25,(#160,#161));
#164= IfcRelFeatureChange ('22No.',#126,'Geometry',$,'Version 2','MODIFIED',#25, (#161,#20));
#165= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(30.),$);
#167= IfcRelFeatureChange ('22No.',#116,' Quantity ',$,'Version 1','ADDED', #25,($,#165));
#168= IfcRelFeatureChange ('22No.',#126,'Quantity',$,'Version 2',DELETED, #25,(#165,$); // A
#170= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(40.),$);
#171= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(50.),$);
#175= IfcRelFeatureChange ('22No.',#126,'Geometry',$,'Version 2','MODIFIED', #35,(#170,#171));
#176= IfcRelFeatureChange ('22No.',# 6,' Quantity ',$,'Version 3',DELETED, #35  ,(#171,$);// B
#200= IFCCARTESIANPOINT((2,3));
#201= IFCPOLYLINE((#200,#199));
#202= IFCSHAPEREPRESENTATION(#43,'Axis','Curve2D',(#201));
#203= IFCPRODUCTDEFINITIONSHAPE($,$,(#202,#195));
#204= IFCBEAM('22No.',#6,'',$,'',#203,$, '3');
#205= IFCPROPERTYSINGLEVALUE('Area',$,IFCLENGTHMEASURE(20.),$);
#206= IFCPROPERTYSET('22No.',#6,'DELETED',$,(#205));
#207= IFCRELDEFINESBYPROPERTIES('22No. ',#6,$,$,(#204),#206);
#210= IFCCARTESIANPOINT((1,3));
#211= IfcRelFeatureChange ('22No.',#126,'Geometry',$,'Version 2','MODIFIED',#204, (#210,#200));
#212= IfcRelFeatureChange ('22No.',# 126,' Quantity ',$,'Version 2','ADDED',#204, ($,#205));
#215= IfcRelElementChange ('22No.',#6,' Quantity ',$,'Version 3','DELETED', (#204));      // D
```

Figure 5.17 IFC-STEP files for model version (M_3)

4. In the model version (M_3),

- The ownership of this model is the same as M_1. Therefore, all the entity instances that represent the ownership of M_0 are returned back to the extended part of the file (#101 to #105).
- Element (D) was deleted in this model. All the entity instances related to this element (#200 to #207) that define the element and all its features are recorded in the extended part. The instance number (#204) for the element is used in *IfcRelElementChange* (#215) to collect the deleted element.

Figure 5.18 illustrates the Change Evolution Graph for the proposed example in addition to the versioning entities and the links between two feature versions. In each model version shown in the graph, the presented versioning entities in that model and in all previous models are required to be recorded in the extended part of that model version. For instance, in model version (2), five *IfcRelFeatureChange* are used to represent the current changes. In addition to one *IfcRelElementChange* and two *IfcRelFeatureChange* from model version (1) and one *IfcRelElementChange* from model version (0), are used to represent the all previous changes.

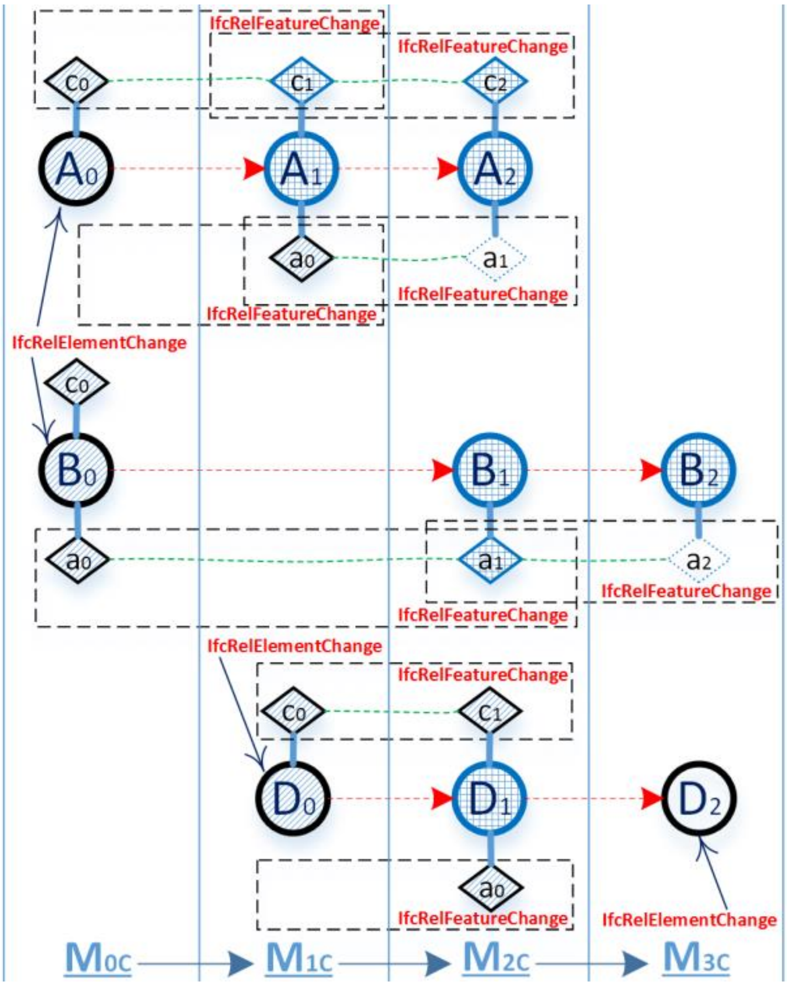


Figure 5.18: Change Evolution Graph for the example

5.4 Summary

The implementation of the collaborative framework for extending the IFC standard was discussed in this chapter. The implementation of the IFC extension was presented in two parts. The first part was to extend the IFC-EXPRESS schema to adopt the versioning approach into the IFC. The main idea was to keep the same structure of the IFC EXPRESS and to define the minimum number of new versioning entities to be used for unlimited number of changes of multi-disciplinary models. The versioning factors were examined and two main versioning entities were described to deal with the model changes in the element and feature level.

The second part was to extend the IFC-STEP file to store the versioning information according to the extension proposed of the IFC-EXPRESS specification. The extension does not alter the current standard IFC and therefore IFC files with the extension added can still be opened by standard BIM software. In the developed IFC file, each element in the model holds its own change history since creation. This improves managing the model and allows tracking the changes that had been made by any designer. An example was presented to explain further the Implementation of the versioning concept in the IFC-STEP file.

Chapter 6

Prototype and demonstration

6.1 Introduction

The previous chapter discussed the first part of implementing the collaboration versioning system, which is the implementation and validation of the versioning concept in the IFC standard. This chapter presents the second part, which entails the development of a collaboration versioning prototype (CVP) to verify the feasibility and usefulness of the versioning concept in an extended IFC model. It also serves as a means for validating the proposed prototype system in a typical design activity.

This chapter highlights the practical issues of the workability of the system. Discussions on the implementation of the prototype are presented in two sections. The first Section 6.2 gives a description of how the collaboration versioning framework is represented in the implementation components. The approach for generating the components is discussed in second Section 6.3. This chapter also outlines the testing and validation of the proposed prototype in Section 6.4.

6.2 Framework Implementation

For the purpose of verification, the specifications for designing a collaboration versioning framework described in Chapter 4 need to be implemented in an appropriate environment. As such, it is very important to select the appropriate design application, sharing service and programming language for the implementation. Therefore, the environment for the framework implementation can be divided in three aspects: (1) BIM software for the creation of design case studies, (2) Programing environment that can be used to develop a plug-in to implement the prototype software, and (3) Distributed median to share the developed model.

(1) BIM software: The AEC environment for carrying out building engineering designs is diverse and has improved in intelligence over the years. The model to represent the building design in this research is the information model (BIM) and the data model (IFC). There are several BIM software applications available on the market today; a few of them can fulfil the requirements for exchanging information using the IFC standard. Although it is possible to link the collaboration versioning prototype to any object-oriented BIM platform (such as Autodesk Revit, Bentley Systems, ArchiCAD, Tekla Structures and others), The Autodesk Revit platform was found to be suitable in this research. It is one of the most widely used BIM applications among users (Gu and London, 2010). It is embedded with an open Software Development Kit (SDK) suitable for such implementation. Revit Platform acquired application programming

interface (API) allows programming with any .NET compliant language including C#, VB.NET or CLI/Managed C++ language.

(2) Programing environment: The history of the computer programming includes hundreds of programming languages. Each has been developed with its relative strengths and unique characteristics. the .NET Platform programming languages (such as Visual Basic .NET, Visual C#, Managed Extensions for C++, and many other programming languages) are intended to be used by most applications created for the Windows OS (Rudder, 2013). Collaboration versioning prototype is mapped and implemented in .NET framework environment using C# Object Oriented Programming (OOP) language. C# is one of the programming languages designed for the Common Language Infrastructure (CLI) and has access to a powerful class library to build components in the .NET Framework (Deitel et al., 2013). C# can interact with the design aspect easily using the application-programming interface (API). It enables programmers to develop applications quickly using an Integrated Development Environment (IDE) found to be highly suitable for implementing the multi collaborative frameworks.

(3) Distributed median: To allow geographically distributed (separated) design teams to share the changed information on a model, a Cloud Computing technology was adopted in this study. Over the past years, Dropbox service has become one of the most popular online file hosting systems. It represents a new generation of file hosting service that not only provides reliable file storage but also enables effective file sharing and user collaboration (Wang et al., 2012b, Quick and Choo, 2013). This on-line storage has been linked with the proposed prototype to export and import the developed IFC file and to share it between the required disciplines based on defined roles in the prototype.

Therefore, the developed collaboration versioning prototype (CVP) system, which was created by using Microsoft C# programming language, was integrated and interfaced with Autodesk Revit through using the Revit API. This enabled the author to gain access to file hosting service operated by dropbox, access the parameter and information of the BIM model, create, edit, and delete model information, automate tasks, perform examination of all elements and features in BIM model, create the extended IFC, and link multi-disciplinary designers together.

6.3 Prototype Implementation

The specifications for designing the framework described in Chapter 4 are used for the implementation of the proposed collaboration versioning prototype system. As aforementioned, the collaboration versioning framework can be sectioned into three distinct workspaces and the processes involved in each workspace is separated into events. UML use-case diagram provides a starting point for clarifying the structure, behaviour, and functions of the system in a simple way (Schneider and Winters, 2001, Yue et al., 2015). It is a list of steps, typically describing the relationships among actors and use cases within a system (Jena et al., 2015). Figure 6.1 illustrates the UML diagram resulting from use-case elicitation of this research. The actors that play the roles in this work are the users of the workspaces (the sender, mediator, and recipient) and the use cases that describe the sequence of events are (modelling, versioning, sharing, tracking, and retrieving).

Use-case is fundamentally different from sequence diagrams or flow charts because it does not represent the order or number of times that the systems events should be executed (Eriksson and Penker, 2000). These events within the use-case diagram need to be more expanded to achieve a good degree of comprehensiveness and clarity in presenting the implementation aspects of this research. The flow chart is pictorial representation of step-by-step illustration of how the events were implemented (Hooshyar et al., 2014). This type of diagram was adopted in this research. A set of flowcharts are presented to illustrate sequentially the implementation procedure of the collaboration versioning process that are adopted by the actors of the workspaces.

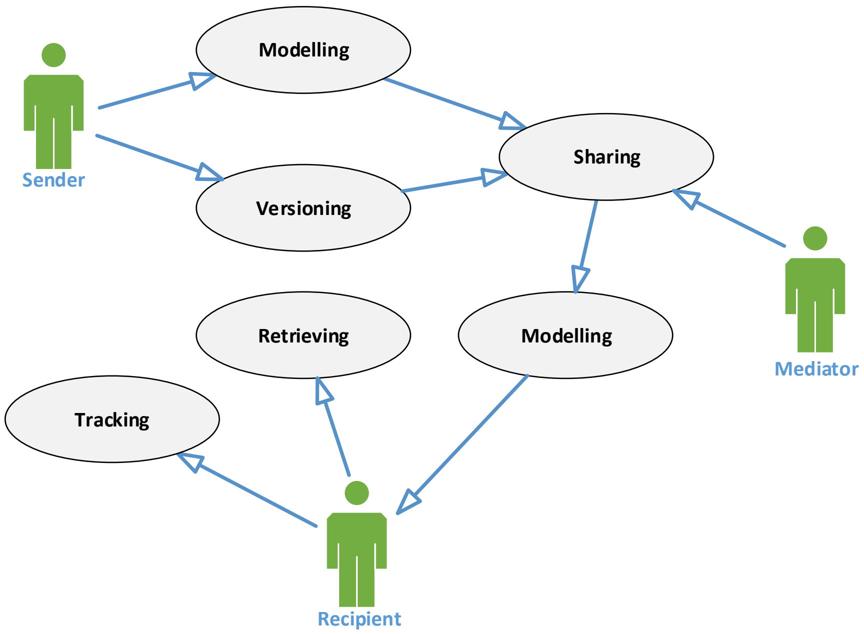


Figure 6.1 Use-Case diagram

6.3.1 Modelling/Sender

The modelling process is shown in Figure 6.2. The overall flow in the algorithm starts when a disciplinary designer (a sender) feels that the amount of maturity and changes in his BIM model is worth sharing with the other disciplinary designers. The next step in the sequence of events is to transform the digital representation of the building (the BIM model) at this stage of the design into data representation through creating an IFC file format from the compiler of Revit. The contents of the new IFC file are the “original part” in the classification that is applied in Section 5.2.2.1.

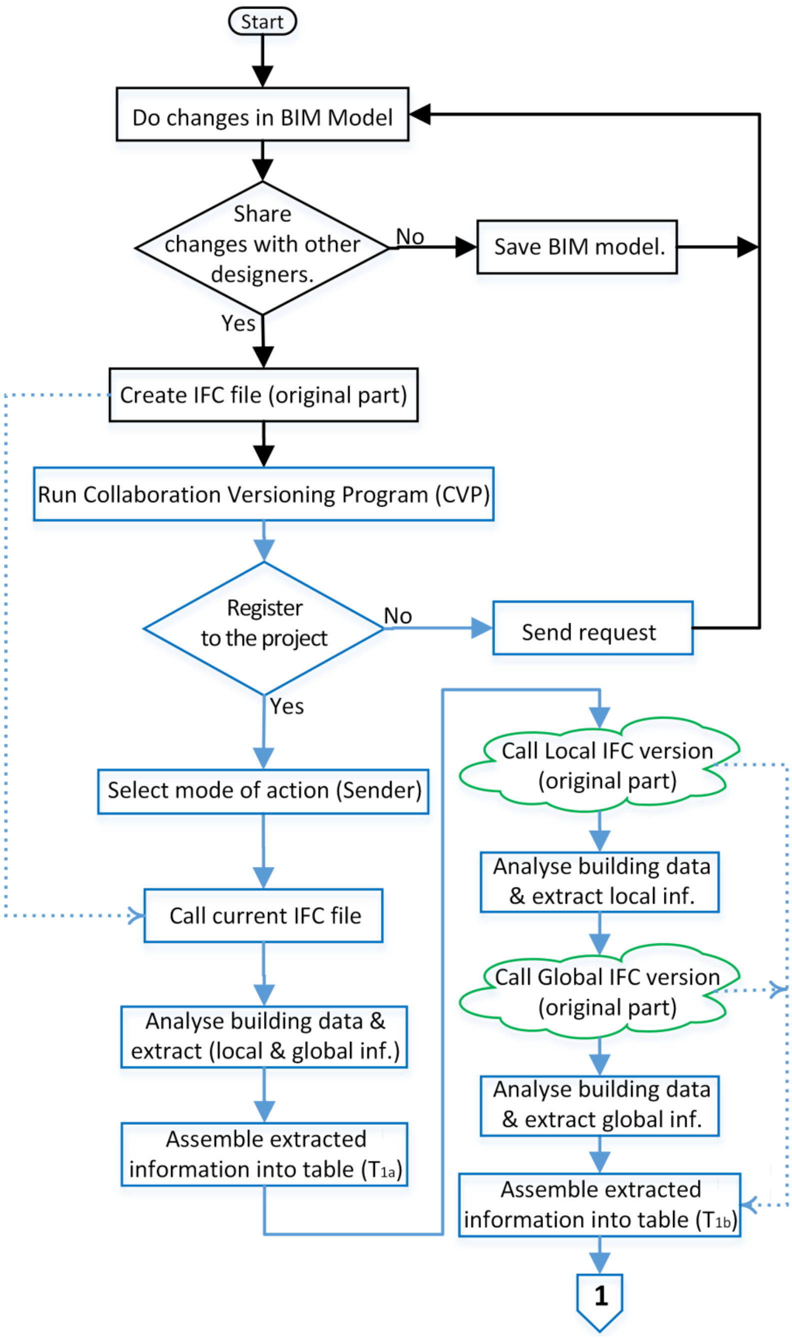


Figure 6.2 Modelling/Sender Flowchart

The sequence of events, as describe in the flowchart, then flows through calling up the collaboration versioning program (CVP) from a Revit application. The program has been interfaced with Revit to run as an add-in tool. The designer can launch it during a building modelling activity through the external link embedded in Revit. The login access is required by the users to operate the program. New users need to get approval and a password from the manager of the intra-disciplinary team, who is a member with the inter-disciplinary team. A registration database is provided to record the information of each inter- and intra-disciplinary designer that involved in the project. A .NET Framework data provider for connecting to MS-SQL database system is used to maintain the database.

The next action is to extract the required information from the IFC files. This includes extract the general information and analyse the physical information. The general information, which related to the current model, is recorded in terms of the creation date, creation time, disciplinary team, disciplinary designer, file name, and application name. The IFC standard is not readily accessible to designers as the structure of the information and their linking is not simple. For instance, a set of entities required to be defined to link the element entity “*IfcBeam*” with the entity that represent the geometry shapes “*IfcIShapeProfileDef*” based on the data schema architecture of the IFC EXPRESS. Generally, Designers do not want to know these linking details and even do not want to know what IFC STEP lines have been changed. The meaning of these changes in terms of the changes in the elements and features are more understandable for the designers. Reading the IFC file line-by-line to extract the information related to the elements and features were discussed in Section 5.2.2.1. Figure 6.3 summarize the process for extracting the elements, IFC-GUIDs, and features information from the “original part” of the IFC model.

Any new IFC-STEP file for any disciplinary team holds affected “global” information and/or unaffected “local” information. The changes of this information in each new model version by any intra-disciplinary team can affect or not affect the other intra-disciplinary teams (as described in Section 4.3.1.1). To manage changes, it is required to study the effect of the change on all disciplinary designers. This determines whether local version and/or global version are required as new model versions. Database change management is proposed (Section 4.3.1.1) to store and retrieve the degree of effect of the changes, in the elements, on multi-disciplinary designers. This is implemented using the MS-SQL Database System within the .NET Frameworks that is used by the Collaboration versioning Program (CVP).

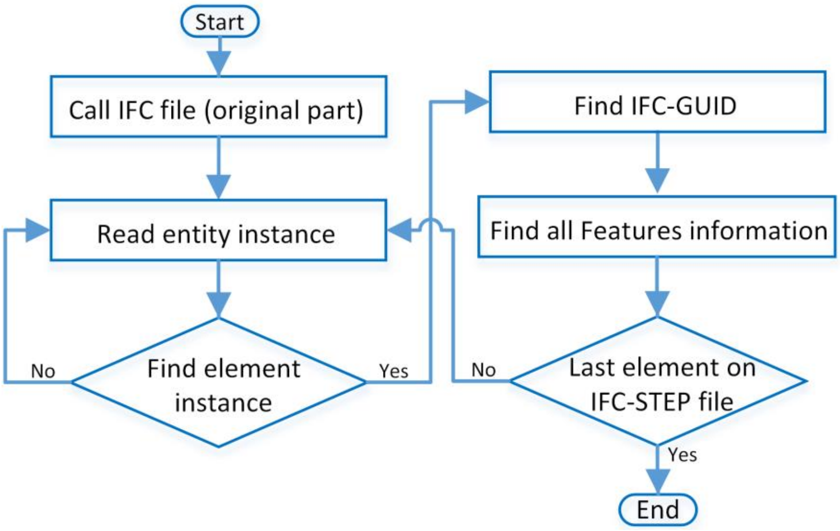


Figure 6.3 Extract required information from the IFC file.

Figure 6.4 shows the information in the current and cloud IFC files. The "original part" of the current (new) IFC file may carry local information that concerns the same designer's team and/or may carry global information that concerns all disciplinary teams. Based on this, the global and/or local information of the current IFC file "c" needs to be collated in Table. This happens through calling the current IFC file, analysing the information in the file, and extracting the local and global information (as in Figure 6.3). The final step is to assemble the extracted information (elements, GUIDs, and features) in table "T_{1a}".

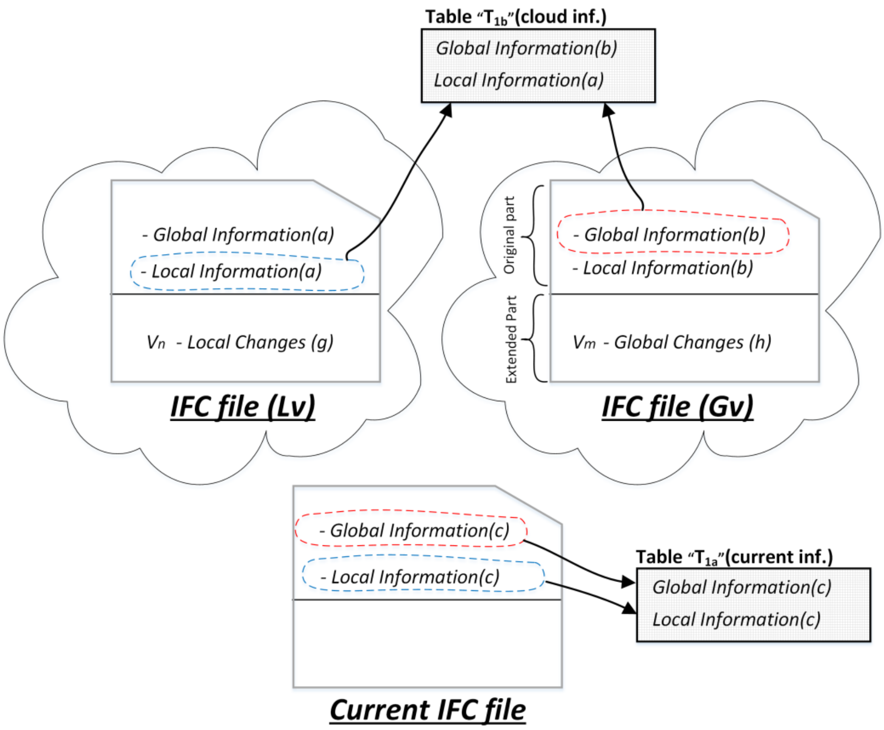


Figure 6.4 Current and cloud information in tables.

The sequence of events of the flowchart (Figure 6.2) then flows through repeating the same steps of extracting the required information for the current IFC file to the cloud IFC files to collect the previous information (local and global information). Local and global versions hold the local and global information of their model version in the “original part”. The “original part” for the global and local files might not be the same because the creation of both versions of the files might not be done at the same time and by the same team (see Section 4.3.1.2).

Since the local version is used with the same intra-disciplinary team, then the developed IFC file will deal with the local information only “a” of the local version file. At the same time, since the global version is used among the same and other intra-disciplinary teams, then the developed IFC file will deal with the global information only “b” of the global version file. Consequently, the process of extracting the required information (Figure 6.3) will be implemented twice to collect the global information “b” of the global version and the local information "a" of the local version in Table "T_{1b}" to be compared later with "T_{1a}".

After creating the latest IFC model and extracting the required information from the current and shared IFC models (the modelling stage), the sequence of events then leads to using the collected information to build the versioning concept in the latest IFC model (the versioning stage).

6.3.2 Versioning

Once the elements and features are extracted and collected in the two tables "T_{1a} and T_{1b}", the sequence moves through managing and tracking the changes between the sequential models by comparing the contents of the tables (Figure 6.5). A similarity-based IFC-GUID and similarity-based feature values for text classification have been used to identify and track all the changes in the elements and features between these two tables.

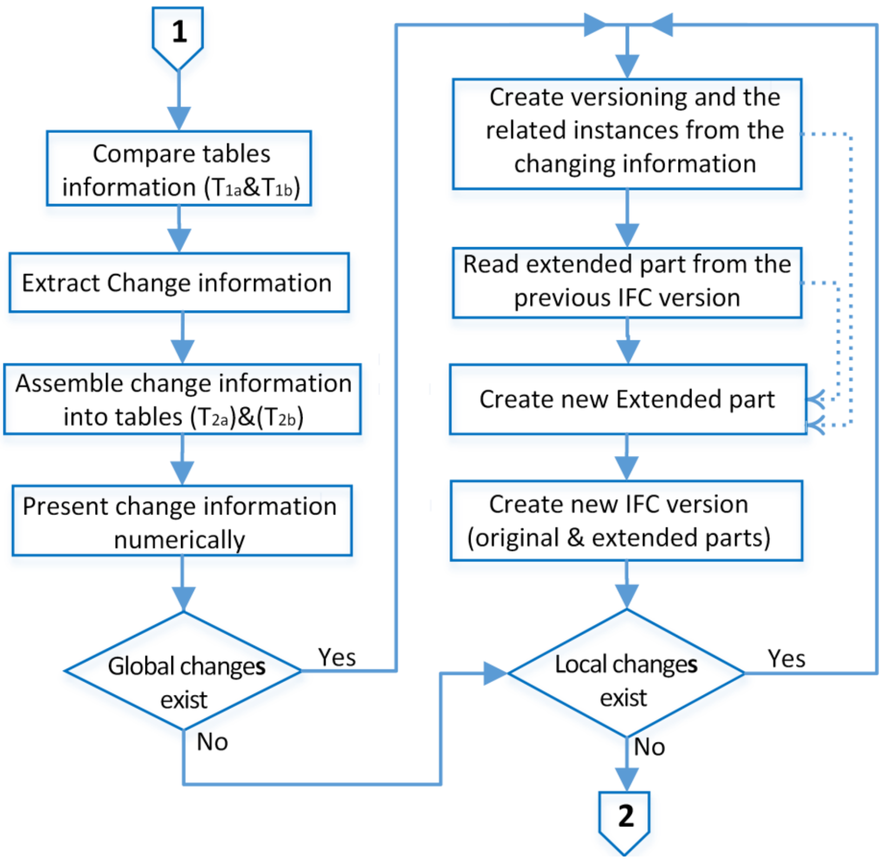


Figure 6.5 Versioning Flowchart

The five-versioning cases of Section 5.2.1.2 (added elements or features, modified features, and deleted elements or features) are illustrated as a flowchart in Figure 6.6. This flowchart clarifies the process of identifying the change types and values of the building information.

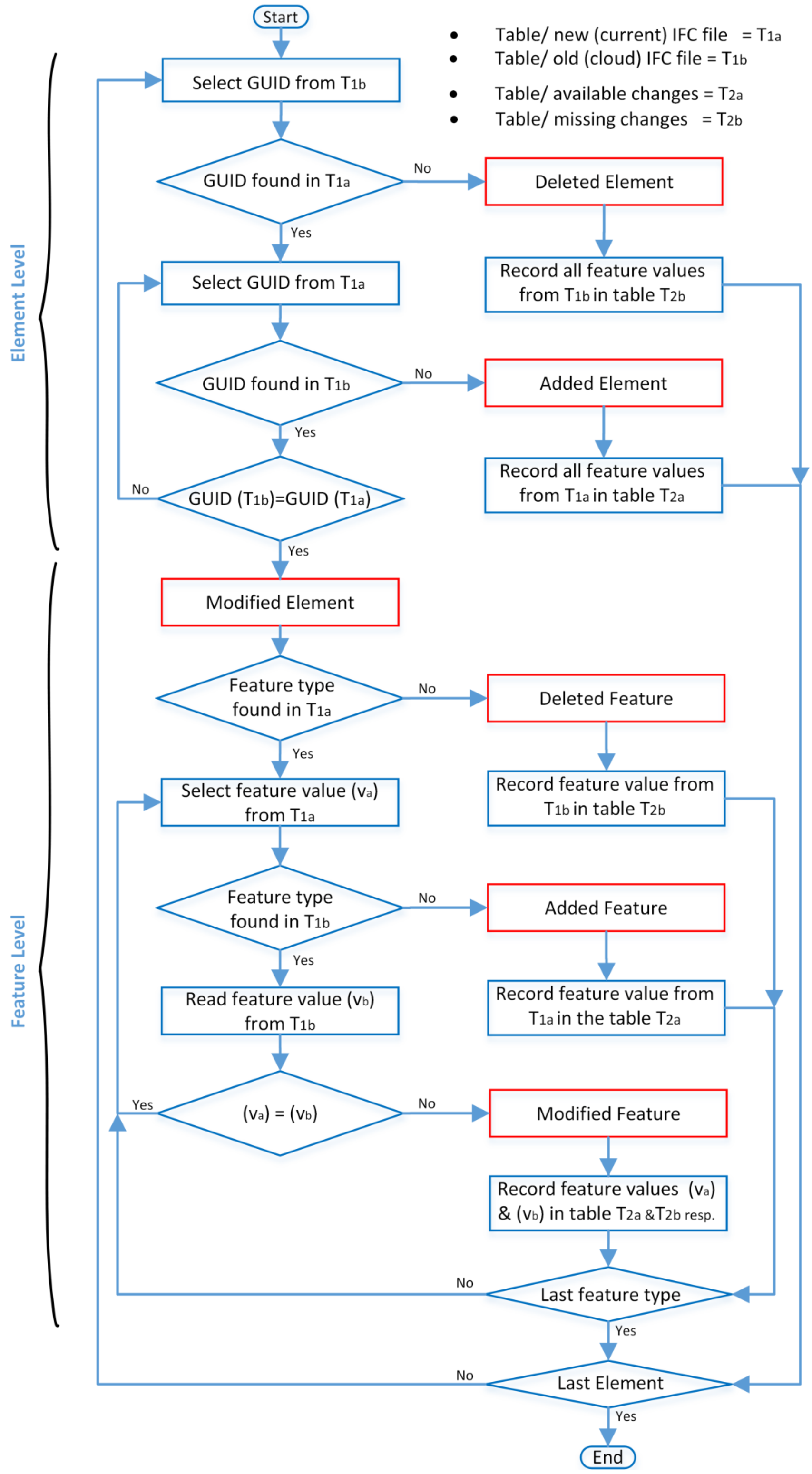


Figure 6.6 Capture changes based on versioning cases (Flowchart)

Based on the comparison results, the added information in the latest model version, the deleted information in the previous model version, and the modified information in the latest and previous model versions can be verified and documented in two new tables for changes. Therefore, the new tables for the available "T_{2a}" and missing "T_{2b}" changes in the model are established. The matching process results in identifying all the changes in the element and the feature levels. Figure 6.7 summarizes the process of collecting the whole and change information of the model in tables.

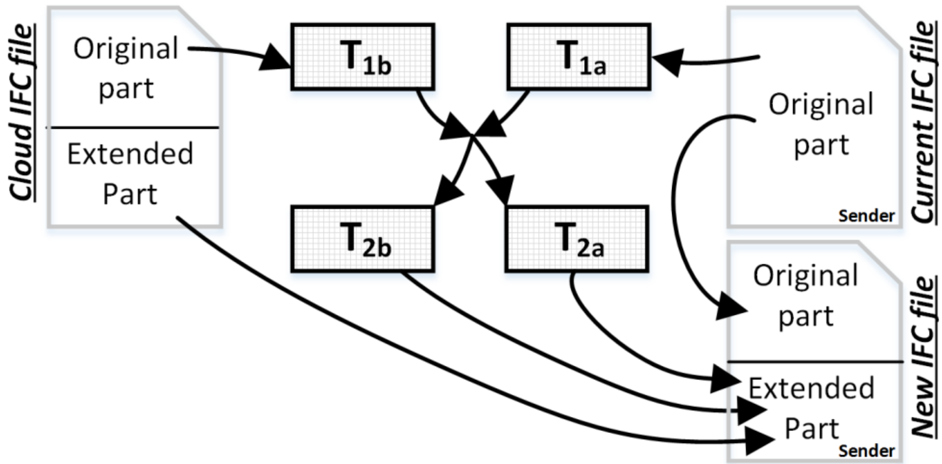


Figure 6.7 Generating tables "sender side".

There are three options for dealing with change information and generating new IFC file(s) when comparing the contents of "T_{2a} and T_{2b}" (Figure 6.8). Option (1) is when the comparison holds local changes only; this leads to generating a new local model version (IFC file) in the local cloud only. Option (2) is when the comparison in the tables holds global changes only; this causes to generate a new global model version in the global cloud only. The last option (3) is when both local and global changes are within the change tables, this leads to generating local and global model versions in the local and global clouds.

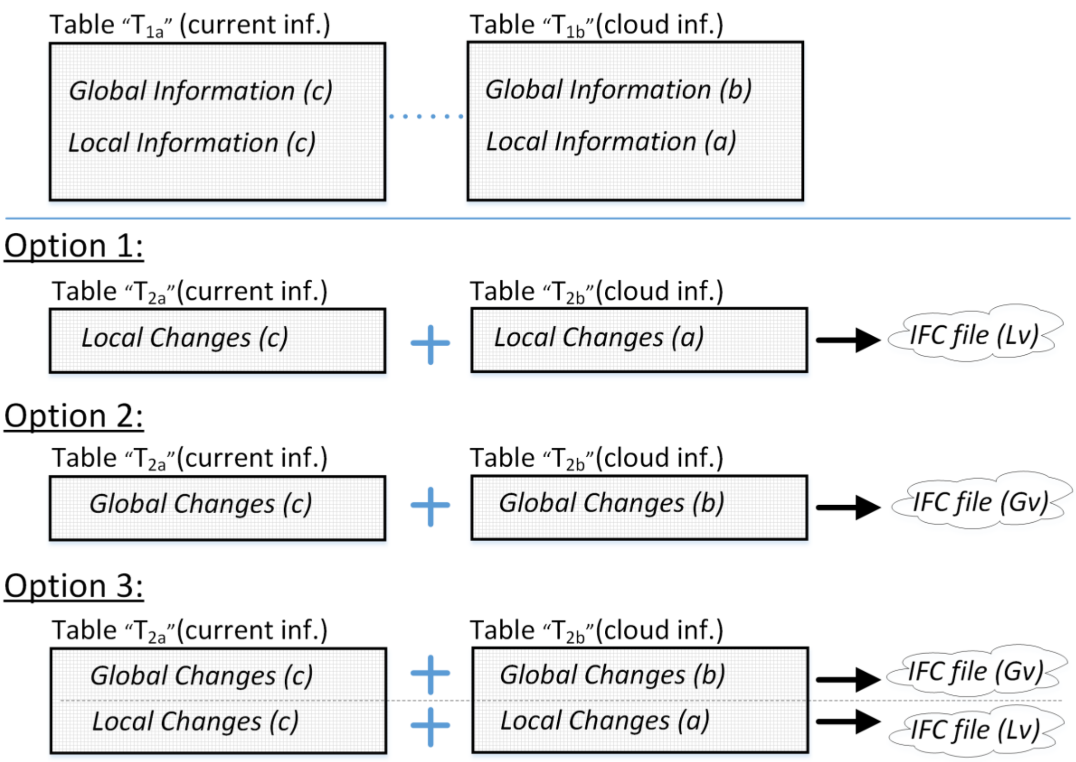


Figure 6.8 Different options to generate new model version in the cloud

In option (3), the contents of the "original part" for the local and global IFC files are the same and completely different for the "extended part" (Figure 6.9). The "extended part" holds the current and history of the local changes in the local IFC file and that of global changes in the global IFC file.

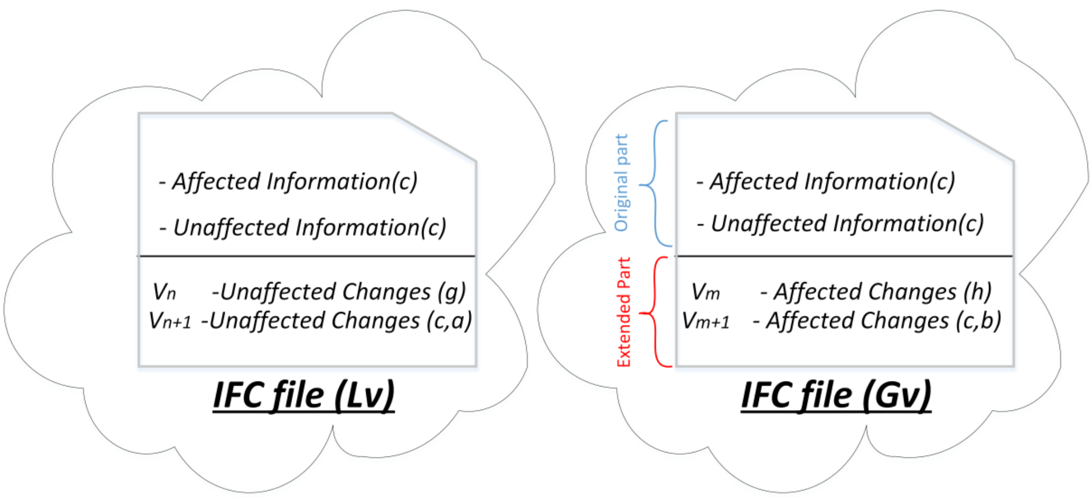


Figure 6.9 Two new mode versions in the local and global clouds

The next step in the process of versioning (Figure 6.5) is to translate the change information in tables "T_{2a} and T_{2b}" into data model language (IFC model) based on the proposed extension of the IFC-EXPRESS schema (Section 5.2.1) and the implementation of this extension in the IFC-STEP file (Section 5.2.2). The process of generation the developed IFC files is the same when there are local and/or global changes. Therefore, the descriptions of the generation process will not mention the change type for the new model version.

The "extended part" of the IFC file contains new and old entity instances, which represents the whole versioning information. The new entity instances are created based on the current changes that are collected in tables "T_{2a} and T_{2b}". They represent the versioning instances, besides the elements and features instances that related to the missing information. Whereas the old entity instances have come from the "extended part" of the existing model version in the cloud, which represent the earlier changes (the historical versioning information) in the previous disciplinary BIM models. The new and the old entity are combined and linked together in the new IFC file. The flowcharts in figures 5.8, 5.9, 5.10, and 5.11 gave detailed illustrations of how to generate, attach, and link the versioning information.

To further demonstrates the "extended part" of the IFC file, the five-versioning cases are represented graphically in Figure 6.10. To avoid displaying lots of information, only a single versioning instance is showing for each versioning case. The current (new) IFC file and the shared (old) IFC file in the cloud are shown in each versioning case to clarify the extracted information from the "original part" of the new and old files and the historical versioning information in the "extended part" of the old file to be added in the "extended part" of the new file.

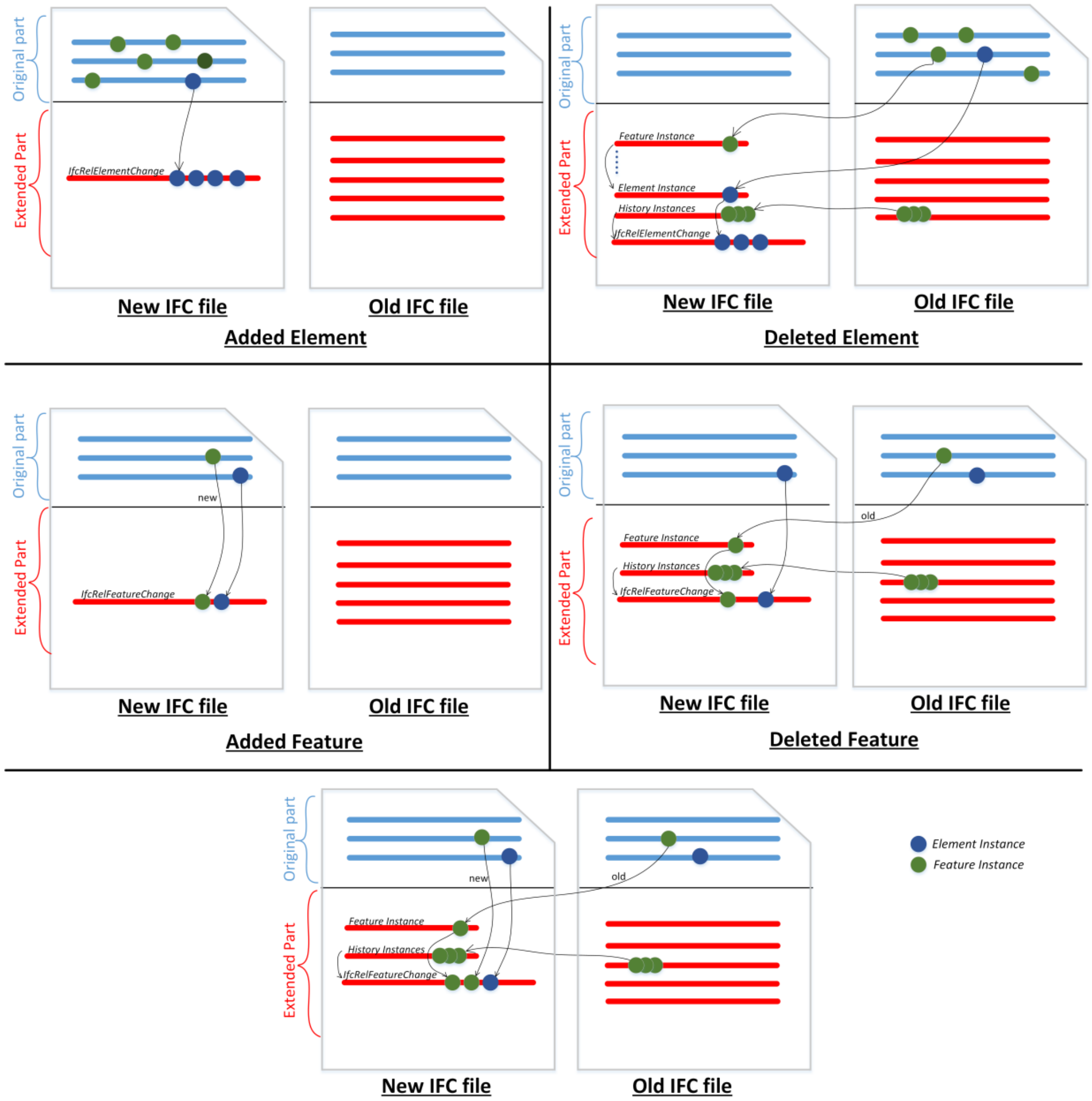


Figure 6.10 Representation of the five-versioning cases graphically

6.3.3 Sharing

So far, in the sequence of event, the developed IFC-STEP file(s) "the original and the extended parts" is ready to be shared with the other disciplinary designers. Figure 6.11 shows the sharing stage. The proposed local and global clouds in Section 4.3.1.2 are used for sharing the new model version. A dropbox service is used to provide cloud storage for the developed IFC file. The sequence of actions commences by checking whether the new IFC version(s) belongs to the local and/or global cloud.

For the local changes, the process starts by removing the current IFC version in the local cloud and replacing with the developed IFC file to present a new local model version among the intra-disciplinary designers (if the new version is the first model to be shared, the file is sent directly to the local cloud). At the same time, notice is sent by e-mail to all intra-disciplinary designers of the sender team to alert them about the new version. The global changes in the current model version need to be shared with the same and other disciplinary teams. The sender then sends the new global version temporarily to the local cloud and simultaneously, he will send another e-mail message to the intra-disciplinary designers of the sender team to make them aware of the new model “global” version. The last event from the sender side is to terminate the “CVP” program.

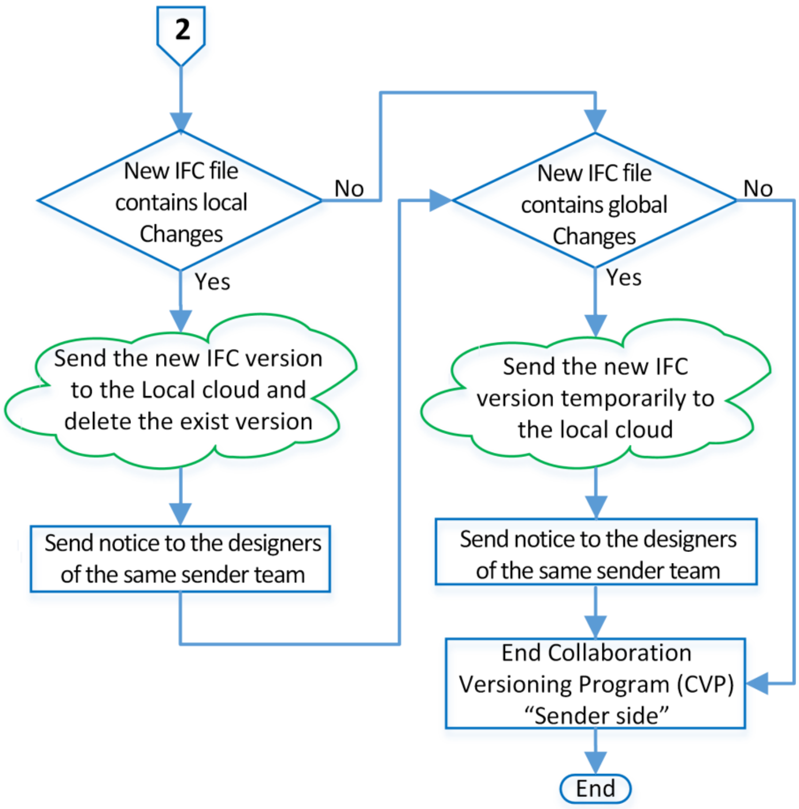


Figure 6.11 Sharing Flowchart

After approving the global changes among the intra-disciplinary designers of the sender team, the new global version then has to move from the local to the global cloud. Based on Section 4.3.1.2, one designer (the team manager) in each intra-disciplinary team is with the team of the inter-disciplinary designers. This designer has the permission to send the new model version to the global cloud and share it with the other disciplinary teams. This will help to increase the work management and efficient collaboration between the design teams.

To continue the sequence of the collaboration versioning process, the team manager for the sender team calls the "CVP" program (Figure 6.12). After the login, the team manager will transfer the new IFC version from the local to the global cloud. Simultaneously, he will send e-mail message to other design teams to alert them about the new global version.

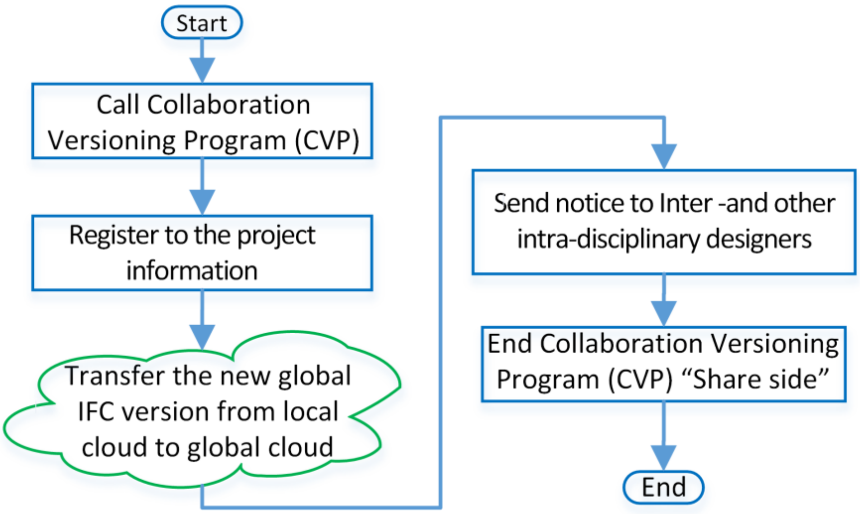


Figure 6.12 Team manager activities (Flowchart)

6.3.4 Modelling/ Recipient

The collaboration versioning process is moved to the recipient designers to read the recent changes from the sender model and all the historic changes from other models and display them in his model. The recipient may be any designer at any of the design teams that involved in the project. He has his own BIM model and works in parallel with the other designers. The modelling process starts when a disciplinary designer receives a message from another designer in his team or from other teams alerting him about the issuance of a new model version in the local/global cloud (Figure 6.13). The steps in the recipients side of creating IFC model to represent the recipient BIM model, calling up of the collaboration versioning program, registering and calling up of the new IFC model follows the same procedure at the sender side.

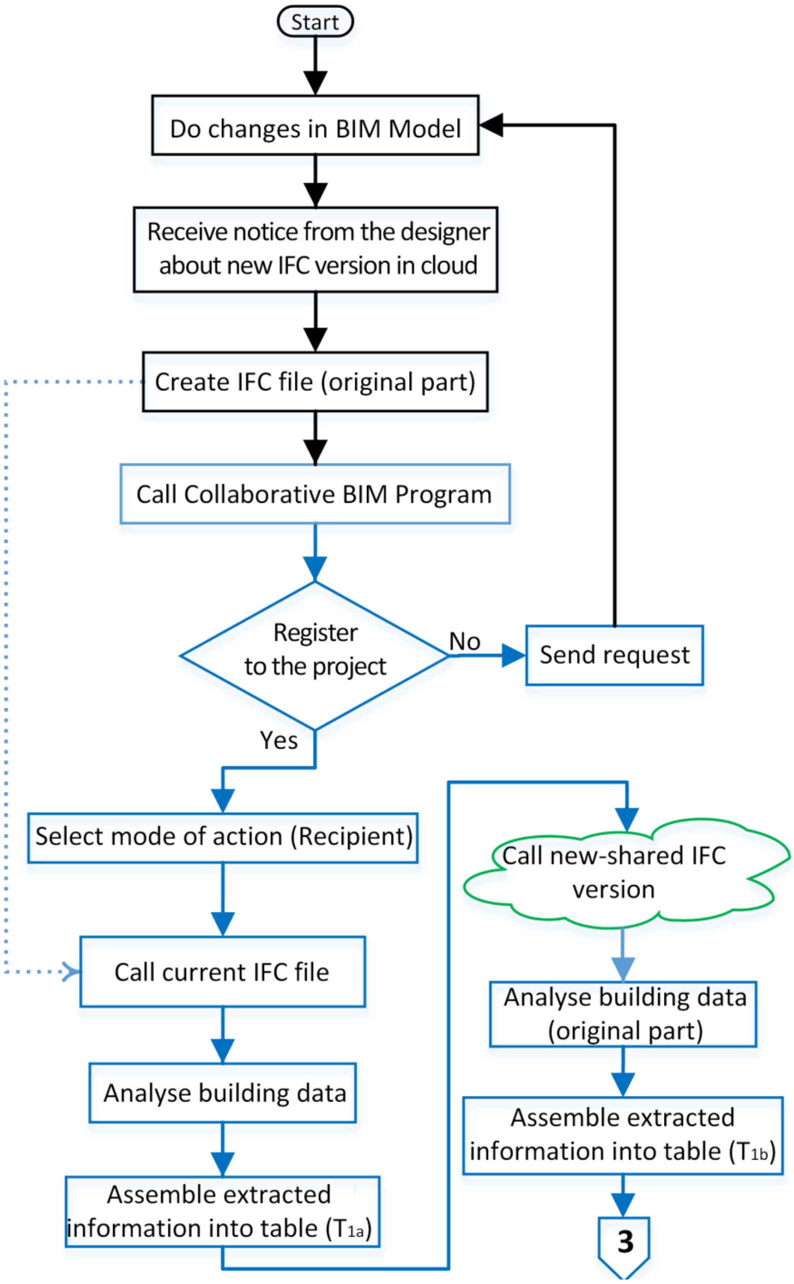


Figure 6.13 Modelling/ Recipient Flowchart

The two model files in the sender and recipient sides are different. The sender dealt with new-updated model and old-shared model in the cloud. These two models are sequential and have been issued at two different times. On the other hand, the recipient deals with new-shared model in the cloud and current-updated model. These are two models evolved in parallel and have been updated at the same time since the last shared model version. Both models hold two different set of change information over the same period. Therefore, the processes carried out on the sender's side for extracting and comparing the information from the "original part" of the two files does not help the recipient to find the latest changes. For instance, if the sender moved a column in his model and the recipient deleted the same column in his model during that time. Comparing the two models on the recipient shows as if there is a new column added by the sender.

The sequence of events then flows through calling the IFC model of the recipient, analysing the contents to extract all the physical and general information and assemble them into table "T_{1a}" (as discussed in Figure 6.3). The last sequences in the process are repeated to generate another table "T_{1b}" from the "original part" of the new model version in the cloud. Next section begins with identifying the changes and clarifying them in the recipient's model.

6.3.5 Tracking

Based on nesting the versioning concept into the IFC file, the most recent with all previous changes have been documented in the cloud IFC file. The latest change information that is required by the recipient can be found implicitly with the "extended part" of the shared IFC version of the sender.

The first step in the sequence of events of the collaboration versioning process in the tracking section is to extract the latest changes from the new model version (Figure 6.14). The sequence starts to call the new model version from the cloud and to search in the "extended part" for the recorded change information of the latest version number. For instance, if the new model version in the "extended part" holds versioning information for (0 to 4) version models, then, version number (4) represent the most recent changes that need to be extracted.

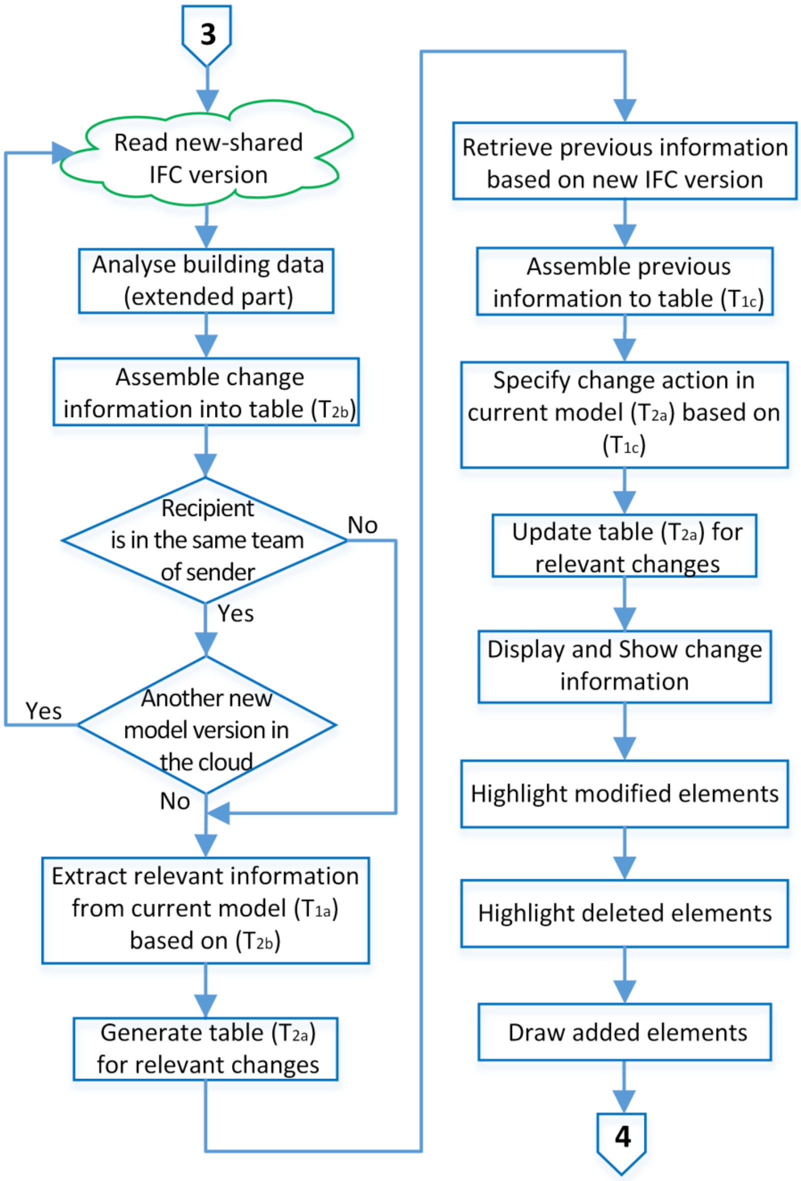


Figure 6.14 Tracking Flowchart

Each of the five-versioning cases is different in the extraction process for the changes. Figure 6.15 describes the process of analysing the new model version and extracting the latest changes. The last step in the sequence of identifying the latest changes is to assemble them into table "T_{2b}".

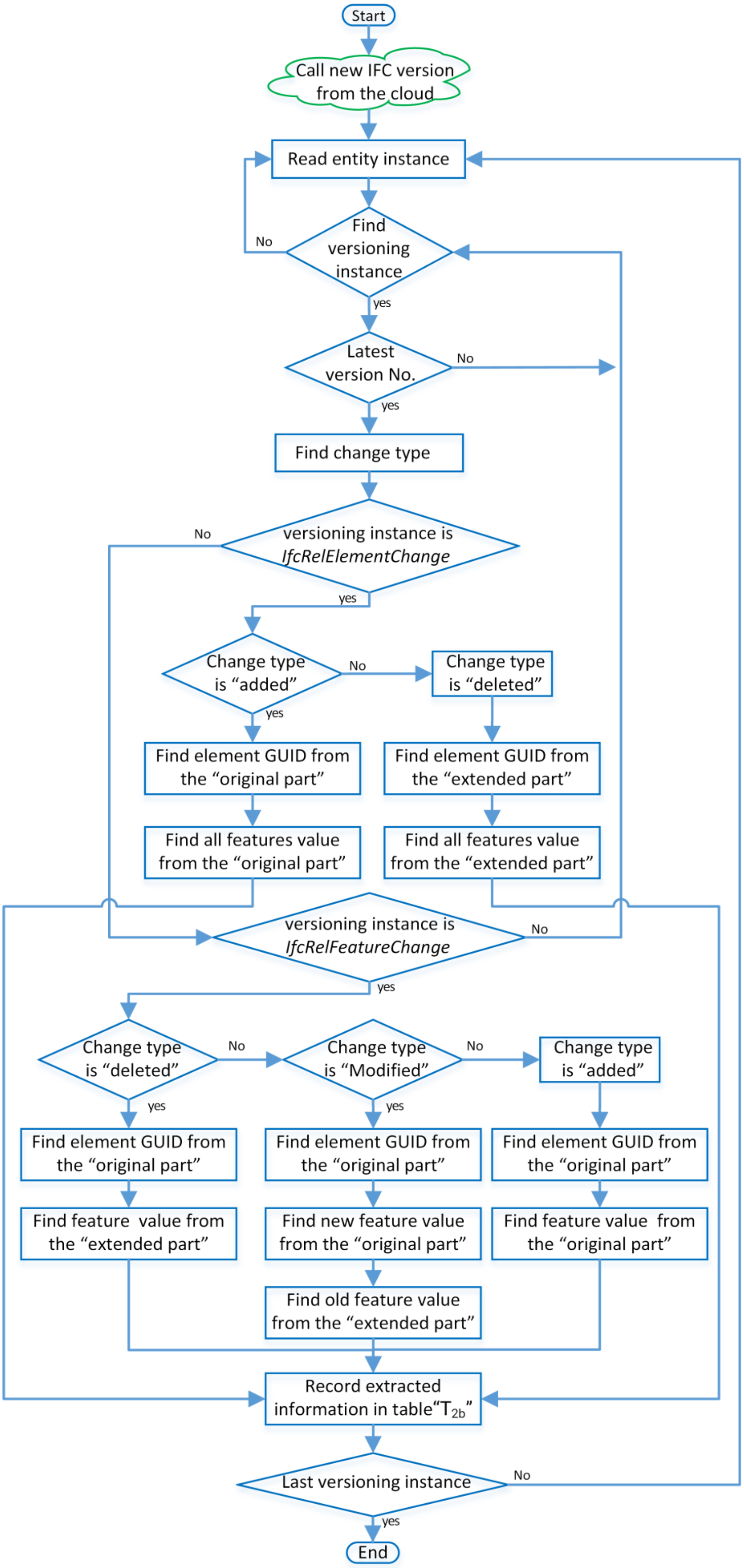


Figure 6.15 Extracting latest changes from the developed IFC (Flowchart).

For the case of two new IFC files are issued in the local and global clouds and the recipient is from the sender's team, the processes for calling the shared IFC file, analysing the file, and assembling the change information into tables (Figure 6.3 and Figure 6.4) are repeated twice. From the ownership of the new cloud file and the login details of the designer, the program can identify the relation between the sender and the recipient to clarify the cloud file(s) that needed to analyse. To establish a comprehensive table for the current changes in the shared models, the new changed information from the second IFC file in the cloud will be added to the table "T_{2b}", which represents the changed information from the first IFC file. Figure summarizes the process of collecting the whole and change information in the tables.

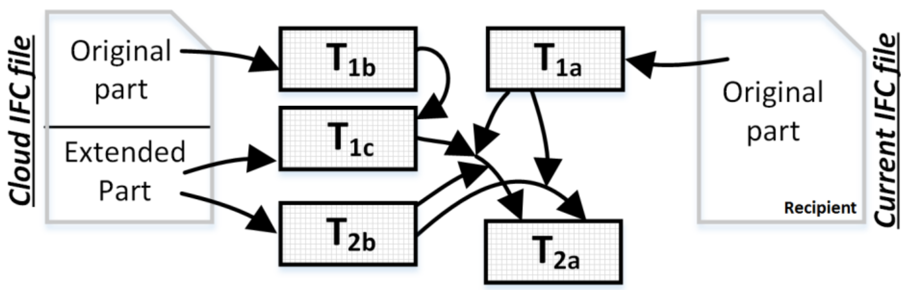


Figure 6.16 Generating tables belong to recipient side.

The next step is to identify the elements and features in the current model that is related to the changes that have occurred on the shared model. From the changed information of the sender model "T_{2b}", the relevant information in the recipient model "T_{1a}" can be identified and documented in new changed information table "T_{2a}".

To show a complete picture to the recipient, the states of the relevant information from the previous model version to the current model of the recipient need to be specified. Therefore, the relevant information in the current model "T_{2a}" needs to be compared with the previous model version. The previous model version ceases to exist after adding the new model version to the cloud. Here comes the benefit of integrating the versioning concept into the IFC file. From the versioning information of the new IFC version in the cloud, the previous (one version back) information about the elements and their features can be regenerated and documented in a new table “T_{1c}”. The process of retrieving the information is further discussed in the next section. After providing information about the previous model, the next step is to compare table “T_{2a}” and “T_{1c}” to specify the change action in each elements and features in the table “T_{2a}” and keep only the relevant changes.

Two illustration types (numerical and graphical) are used to manage the changing information more clearly. In addition to providing the change information in tables, each changed element in the new and previous model versions and in the current BIM model can appear separately to compare the feature values between different models numerically. The CVP has been designed to facilitate easy human interpretation of the information contained in the building models. The graphical representation of the design change is essential for engineering design systems. The prepared visualisation feature enables the designers to understand the building changes and to visualize the sender changes through the recipient model. The recipient needs to illustrate the changed information that has been sent from another designer on his model. This process further depicts the change information by highlighting the modified and deleted elements and drawing the added elements. The Revit Platform API provides a mechanism for filtering and iterating elements in a Revit document through using family instance of elements that can programmatically access this path of family directly and load the needed information (Autodesk, 2013). This way, editing functionality in the

Revit Platform API can be used to highlight the elements specified. Added elements can be drawn through finding the family instance of the element type based on the provided features information about the element.

6.3.6 Retrieving

Having a method for adequately capturing and storing information is only useful if that information can later be retrieved by any user in a method that can effectively support their work (Wang et al., 2012a). The historical information that has been stored within the developed IFC file can be retrieved and reused by any discipline. This aspect is optional when the recipient wants to obtain historical information of the changes in the project model versions. There are occasions where it would be necessary to revert the model back to earlier stages of design. Examples of this include cases of unresolved conflicts or the need to view the evolution of information on the model in a chronological order. For that purpose, the retrieving aspect in the collaboration versioning process allows for the restoration of the design data model to prior states. This version control mechanism of the model allows the system to retrieve any earlier state of the model. Retrieving the old information might include the changes at three different levels of information based on the structure of the building information in the model. The classification of retrieving the old information can be:

- At the feature level: it includes the change evolution of a specific element feature in all model versions, such as knowing the changes in the geometry shape for a particular beam.
- At the element level: it includes reviewing all changed features of an element from creation to current state. Like the development of a beam from its creation until the current version.

- At the model level: it covers restructuring of the current model with all its elements and features to return to an earlier model version at a specific time. As in reshaping the model version “three” from the current model version “five”.

These three levels will be the basis for retrieving the required single feature, element, or model from the current model version by relying on the version history within the developed IFC file.

The first step in the sequence of event of the collaboration versioning process in the retrieving section is to select one of these three levels. Figure 6.17 shows in a simple way the retrieving flowchart and clarifies the three levels. The next step is to call the current model version from the cloud to deal with the required information. The procedure for filtering the contents of the IFC model version and extracting the required information for each level is discussed in more detail in the next paragraphs. After extracting the information, the results are shown as a table with the program and viewed on the Revit screen. Eventually, the last event in the sequence is terminating the “CVP” program from the recipient side. Information about element or feature lifecycles, which includes all the changes from its creation until the deletion or the last modification, is available information in the developed IFC model. To retrieve the history of changes that occurred upon any feature, the sequence of events begins with selecting the feature type, and the element to which the feature is belongs. All the change history for this feature can be extracted from the extended part of the IFC model. The idea is to read the changes in the model versions starting from the newest (M_n) to the oldest version (M_0). Within each model version, the feature information is extracted whether it had been stored within the changes in this version. Figure 6.18 shows in more details the flowchart of displaying the change history for a specific feature.

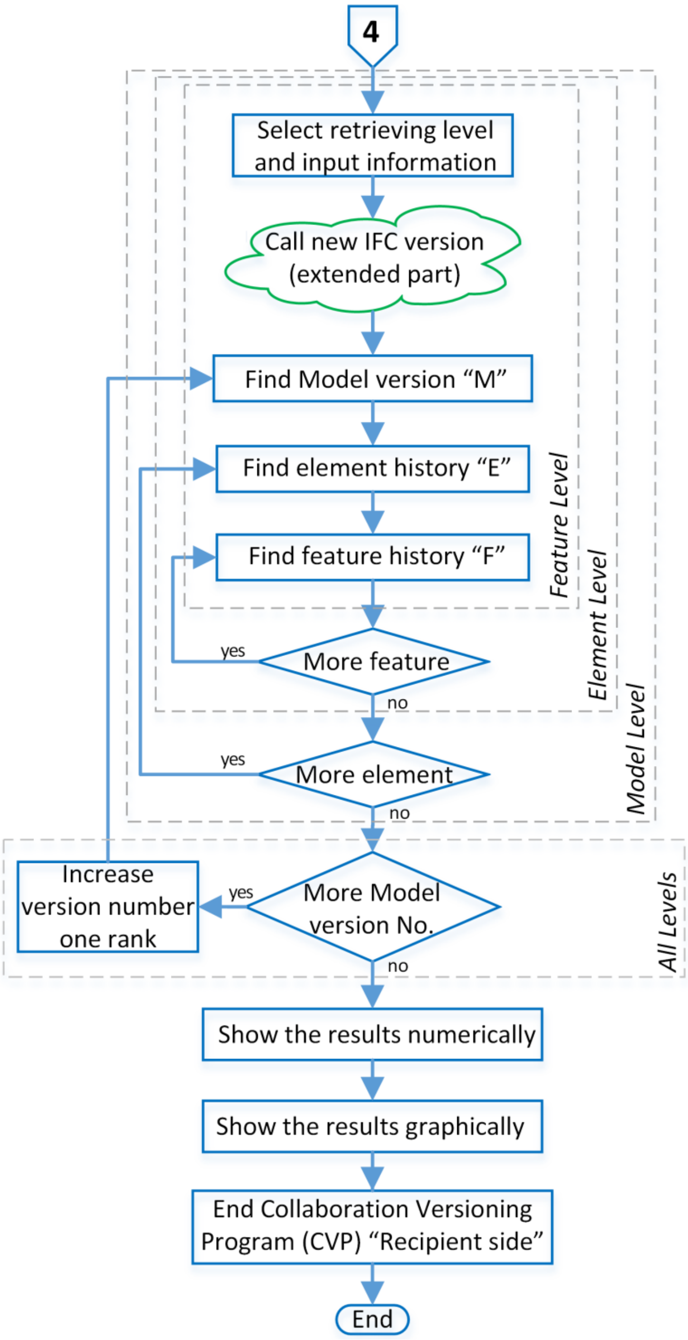


Figure 6.17 Retrieving Flowchart

The process of retrieving the information about a specific element is almost the same as in the feature but on a larger scale. The process needs to identify every single change in the features of that element in all model versions. In the end, the evolution of change of the element has been gathered in tables as an output similar to the change evolution graph that has been introduced in Section 4.3.2.5 and Figure 4.20. Figure 6.19 shows the flowchart of the change evolution of an element. The process of retrieving a particular model version is completely different from the above two types (retrieving information about element and feature). The process needs to retrieve the changed and unchanged elements and their features for a specific version. This full information for any model version can be re-built from the current information in the original part and the history information in the extended part of the IFC model. Since the current model is the only model shared among designers, reading all the elements and features information in the "original part" of the current IFC model and recording them in table is the first step.

The next step is to read one by one the changed information in the "extended part" of the current IFC model starting from the newest to the required version and update the table based on the changed information. Each deleted element or feature between the two above versions needs to be added into the table. On contrast, the added element or feature is unavailable information in the required version and needs to be deleted from the table. For the modified feature, the current value in the table is replaced by the old value of the modified feature in the "extended part" of the IFC model. At the end, the updated table based on the changes of information between the newest and required versions represents the required model version. The results are similar to regenerate element for all the preceding versions that were introduced in Figure 4.21 to reform the other elements at that model version. Figure 6.20 shows, in more details, the flowchart of retrieving a particular model version.

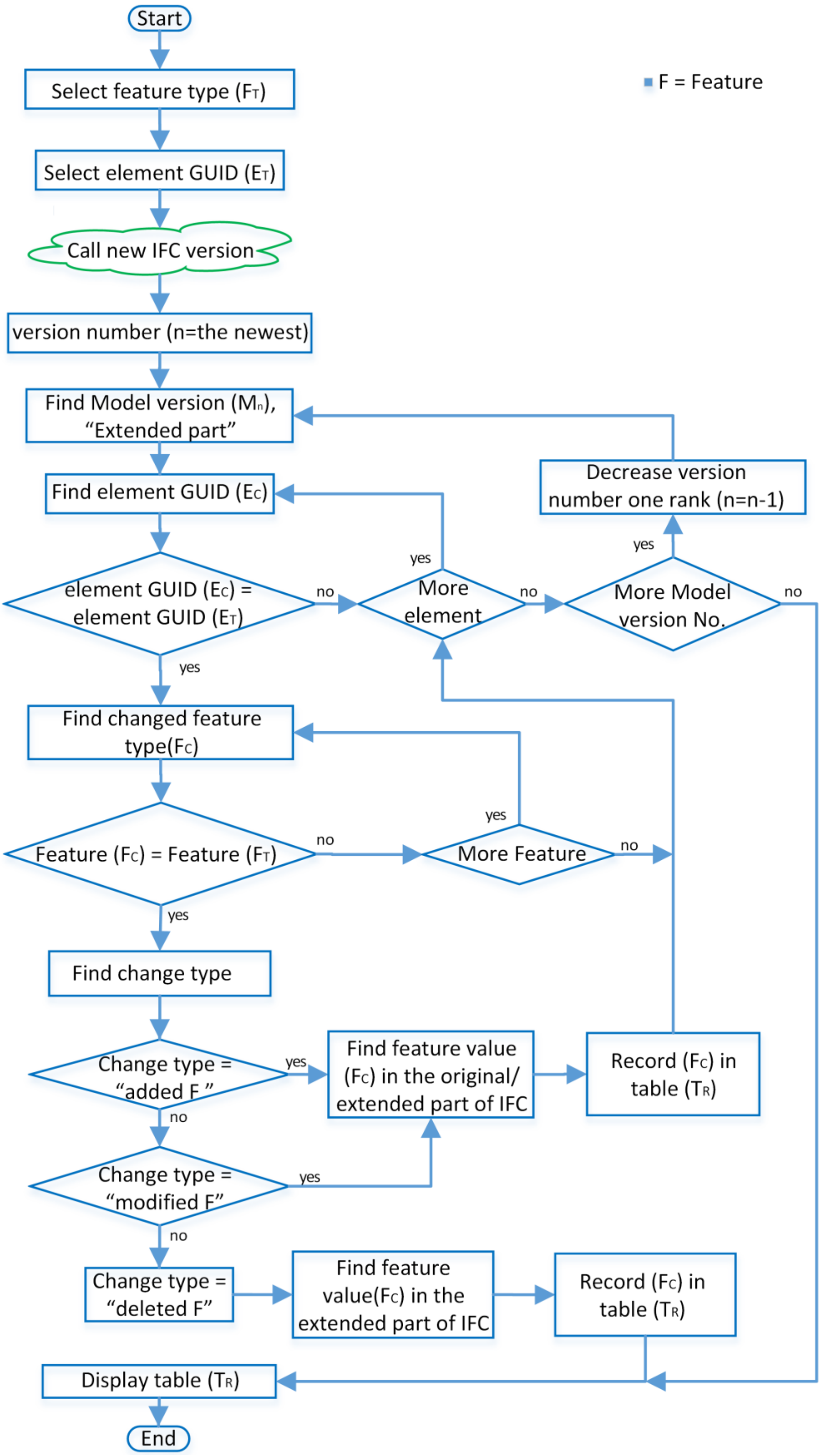


Figure 6.18 Retrieving history of changes at the feature level (Flowchart)

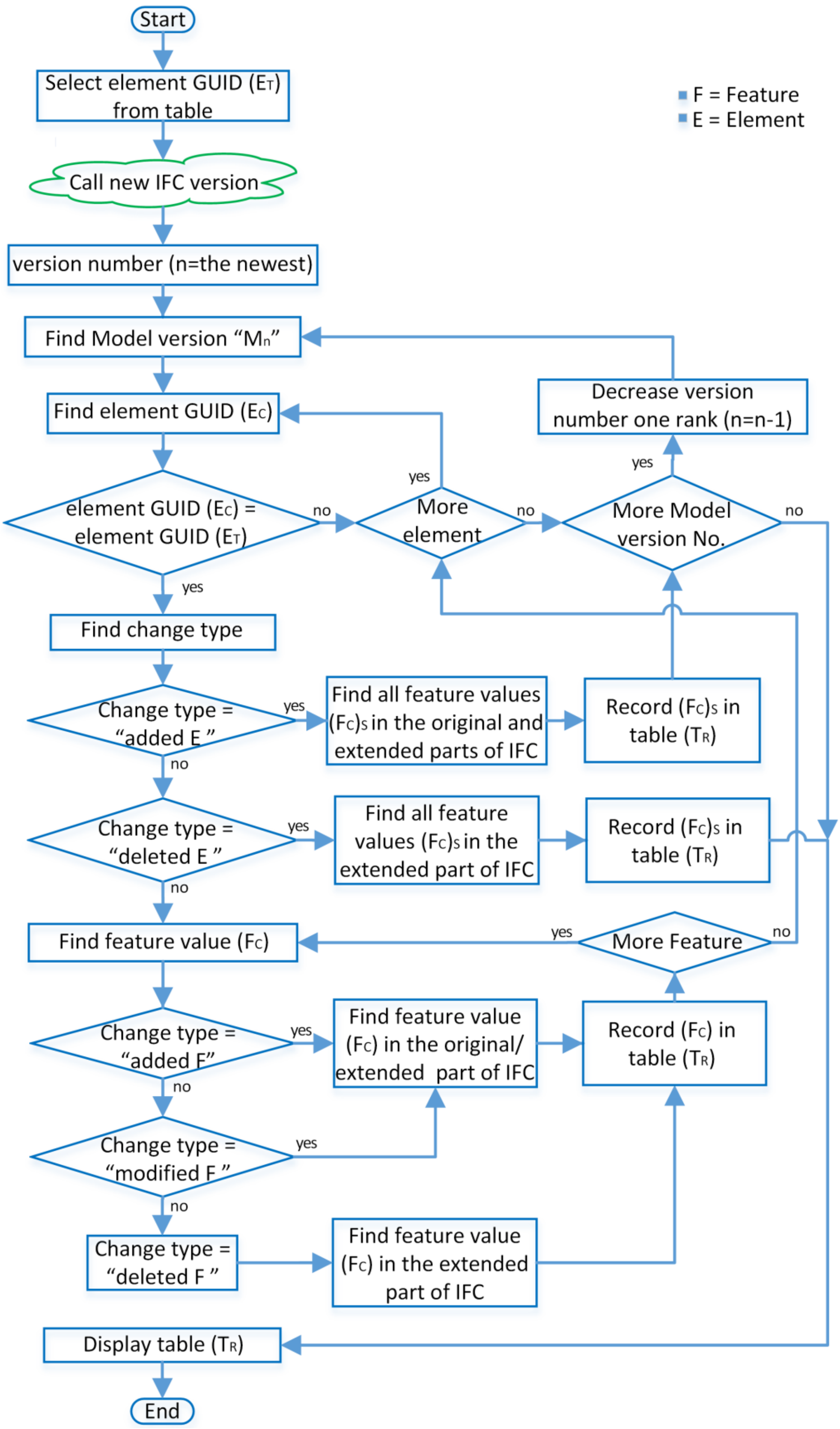


Figure 6.19 Retrieving history of changes at the element level (Flowchart)

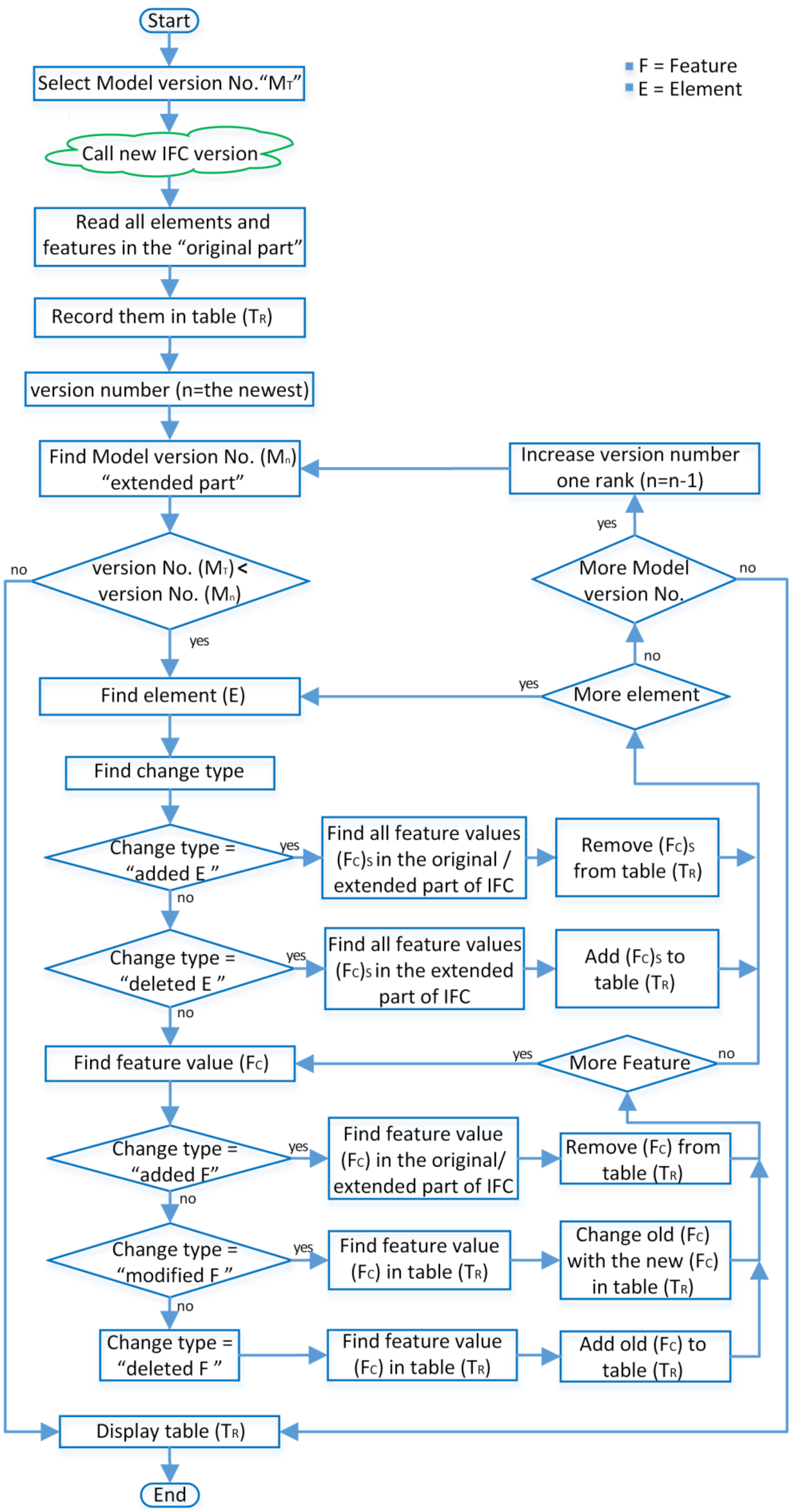


Figure 6.20 Retrieving history of changes at the model level (Flowchart)

6.4 Demonstration Example

The use of the proposed prototype system is demonstrated and validated in a typical design activity. The intention is to illustrate the usefulness of the system in applying the concept of versioning in the design process. An example based on multi-disciplinary designers and multiple-model versions is examined in this section.

6.4.1 Example rationale

An example is used to help present the usefulness of the prototype. The proposed solution uses a realistic example for the purposes of providing significant evidence for the effectiveness of the proposed solution that can be further supported by an evaluation (Zave, 1997). The example in this research follows this line of discourse. The prototype has been used to analyse typical example of building design in preparation for evaluation presented in the next chapter.

The proposed example is designed to illustrate the overall functionality of the developed IFC standard and the collaboration versioning system. This will help to examine how the requirements for extending the IFC standard and for implementing the prototype as well as the collaboration versioning framework have been correctly implemented. The research goal of the example is to investigate and verify the effects of applying the versioning concept in the extended IFC model on the design process to improve the collaborative design. This includes examining the following points:

- Identifying any new, deleted and modified elements in each building model
- Geographically dispersed team members from different disciplines to work together.
- Managing the project access for different designers

- Participating multi-disciplinary teams and multiple designers within each team in the design process of the project.
- Using different shared and specific building elements.
- Sharing a central IFC file (model version) in the local/global cloud among disciplines.
- Explaining all the changes of the BIM model in the extended IFC file by the sender side.
- Tracking all the change history in the extended IFC file on the recipient side.

6.4.2 Implementing the example

To describe the various aspects of the example implementation adequately, overview of the example that is adopted, description of a series of design process that is proposed, and input and output of the CVP that are developed, are presented here. The operations of the prototype with the aid of corresponding screenshots at various stages are also described.

6.4.2.1 Overview of the example

Traditionally, the building design is a complicated process; it involves multi- disciplinary design teams and multiple designers in each team. Each disciplinary team tends to work with its BIM model depending on the speciality of each one of them. The process of creating and changing the single model and sharing it among the designers is the same whether it happening between two disciplinary teams or more or between two designers in the same team or more. Due to the complexities involved in considering all the disciplinary design teams of the project in the proposed scenario, only two intra-disciplinary teams, architectural team (A) and structural team (S), are considered in this example. Each intra-disciplinary team has two designers (A_1, A_2 and S_1, S_2); one of the designers in each is also with the inter-disciplinary team (A_1 and S_1). Therefore, the designers involved in this example consist of a design manager and another designer in each of the architectural and structural team.

The project used for the case illustration is a steel framed building. The principal elements of a typical framed buildings comprise floors, beams, and columns (Zalka, 2012) (Figure 6.21). Some other building elements such as foundation, doors, windows, etc. are included in the drawing of the models. To avoid displaying the frequently used information in the CVP, which represent the same ideas, beams and columns are used as shared information between the architectural and structural models, and foundation element is used as a specific element for the structural team.

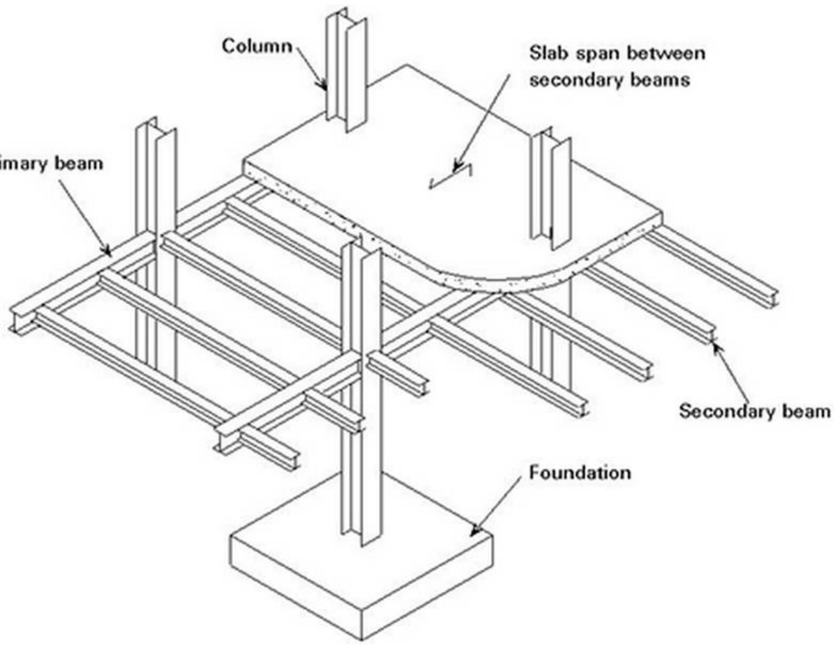


Figure 6.21 The principal element of a typical multi-story building (Zalka, 2012)

The example is based on having a BIM model for each discipline. The two teams can issue different sets of model versions that can be shared in the local cloud for each intra-disciplinary team and in the global cloud for the whole teams, depending on the proposed changes in the BIM models.

The first BIM model "the conceptual model" is created by the architect team, as mentioned in Chapter 2. Figure 6.22 illustrates the 3D-plan and two views of the conceptual model. All the dimensions and elevations are shown in the drawing. The other disciplinary designers build their models based on the conceptual model and frequently revise their models later according to the design requirements.

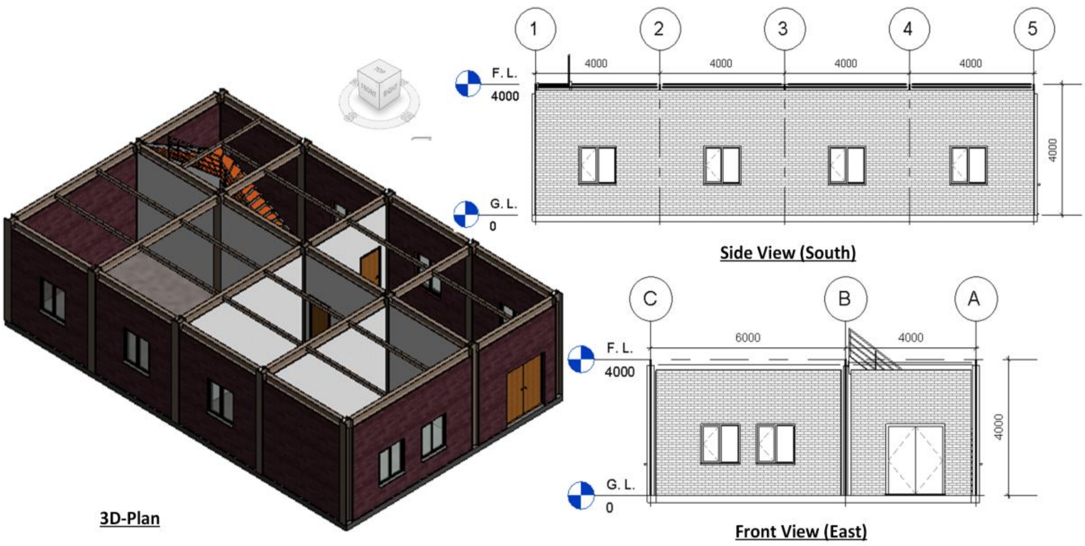


Figure 6.22 The conceptual model

6.4.2.2 Design process and operation screens

A series of design steps have been proposed in Figure 6.23 to explain the creation of two groups of local and global model versions. These steps are described later within the context of the design process (referencing to the figure has been done in the later explanations by writing the step number, such as "Step 2"). Every designer participating within the design process repeats a number of the design steps each time. This leads to obtaining different design scenarios, and generating different types of changes and several model versions in the local and global clouds.

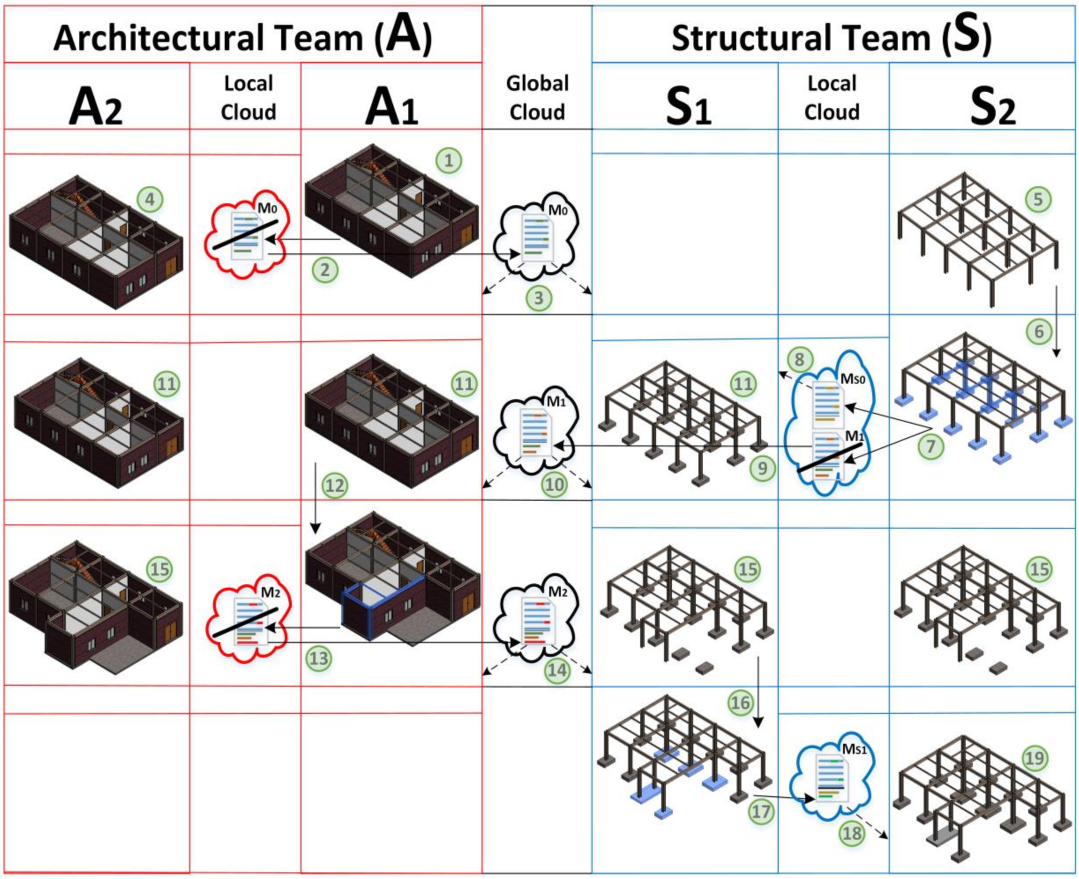


Figure 6.23 The proposed design steps

As the design progresses, each designer (sender or recipient) needs to run the Collaborative Versioning Prototype (CVP) several times. The steps of running the prototype are summarized in Figure 6.24 and will be described more precisely later (the sequence of executing the program steps are indicated by letters).

The process of feeding information into the prototype and checking corresponding output results goes through several operation screens in the sender or recipient sides. These operation screens have been developed based on resulting implementation tasks throughout the research work.

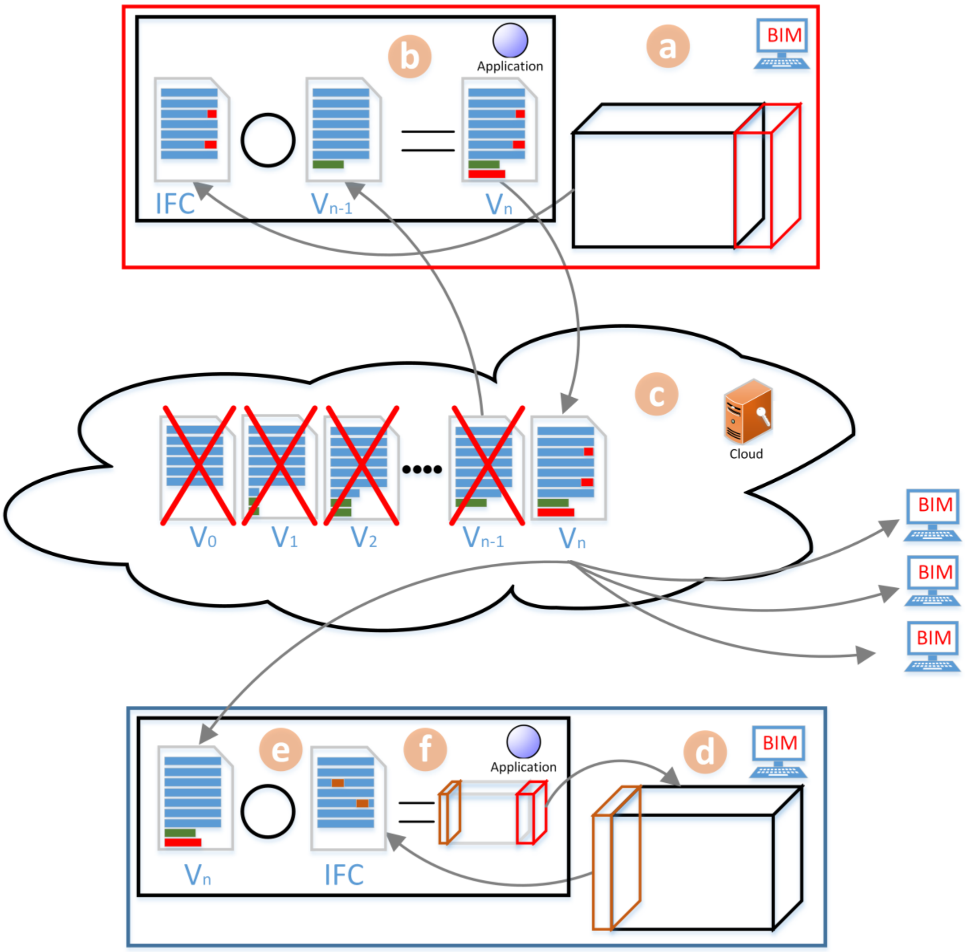


Figure 6.24 Operation of the prototype

6.4.2.2.1 *First Steps*

The design process starts with creating the conceptual model by the architect (A_1). Figure 6.22 illustrated the 3D projection and two views of the conceptual model. To generate more model version files in the cloud, to obtain all the change types (added, modified and deleted), and to deliver the best explanation of the process, the first steps in the proposed process (steps 1 to 11), and the steps of running the CVP (steps a to f) are briefly described according to their respective functions. The design steps that follow step 11 are explained in details in the next section.

The first steps of the design process include, creating the conceptual model (step 1), running the CVP, generating and sending the first model version “M_0” to the local and then to the global cloud (step 2), and sending messages to all designers (step 3). Since all the elements in the conceptual models are new, based on the proposed extension for IFC, only one new versioning instance in the model version is required to collect the new elements. Below is the new versioning instance that has been taken from the IFC-STEP file (version 0):

#3874=IFCRELELEMENTCHANGE('1eCkg00Xf4ahu$slF8zg$x',#52,'Geometry',$,'Version 0','ADDED', (#489,#588,#653,#718,#1137,#1202,#1267,#1332,#1561,#1626,#1691,…….));

To build the registration database, all the designers that will participate in the project needs to fill out a request form in their first access to the CVP program to get a permission to access the program. Architect (A2) in Figure 6.25a is an example to send the access request to the team manager for the architect group (A_1). A_1 later will add the new designer’s information to the system and send reply e-mail to A_2 contain the access information to the CVP. Figure 6.25b shows this step.

After sharing the first model version among the designers, the architect (A_2) reviews and saves the architectural model (step 4) and the structural designer (S_2) creates the first structural model based on the architectural model (step 5). The process of accepting or rejected the new changes by the recipients are out of scope of this research. The structural designer (S_2) then starts to develop and analyse the structural model by adding the foundations and changing the sectioned shape of some columns and beams (step 6). The same process mentioned above is repeated for (steps 7-11).

At step 7, two new model versions are created. The first is the first local model version “M_{S0}” that is shared between the structural designers. The “extended part” of the new model version “M_{S0}” contains information about the specific structural elements only (the new foundations). The second new model is the second global model version "M_1" that is shared between the structural team only in the local cloud to discuss and get the final approval about the shared building elements. An alert message is sent to (S_1) about the new model versions in the local cloud (step 8).

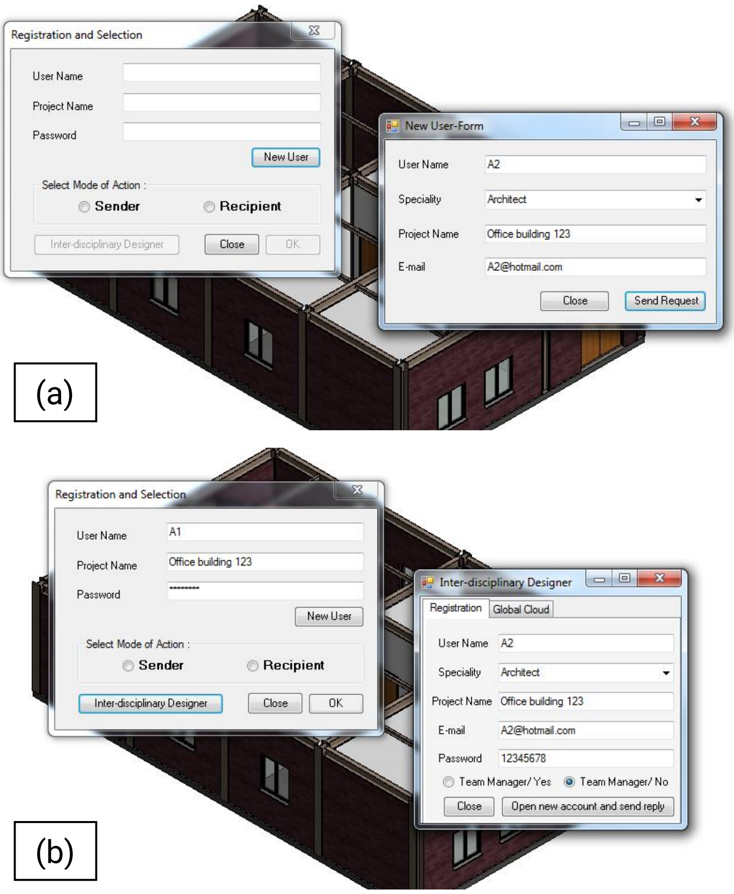


Figure 6.25 Registration Process (a) sending access request to team manager and (b) Adding new user to the system.

The designers are granted default permissions based on their respective roles defined within the prototype (Section 6.3.1). Only the team managers have this authority to add and delete the version file in the global shared folder and send messages to the other disciplinary teams. In the current example, the structural designer (S_2) does not have the permission to send the new model (M_1) to the global cloud. Therefore, the team manager (S_1) then share the new global model version "M_1" from the local structural cloud to the global cloud that all disciplines can read the file (step 9). The old model version "M_0" has been replaced with the new one "M_1". The "extended part" of the new model version "M_1" is about the shared information only (the beams and columns). An alert message is sent to all designers to remind them about the new model versions in the global cloud (step 10). The step of running the structural manager (S_1) the CVP to transfer the new model from the local to the global cloud (step 9) will be discussed later.

At this stage of the design process, two versions of changes are recorded within the shared global model "M_1" and one version is recorded within the shared local model "M_{S0}". The designers (A_1, A_2, and S_1) then updates their models based on the new model versions (step 11). The following section discusses the ensuing steps in the proposed process, and the steps for running the CVP in detail.

6.4.2.2.2 Descriptive steps

Having all disciplinary designers been working simultaneously, the architect (A_1) has carried out some changes on his model that he wants to share with others (step 12). The architect can do so by generating the IFC file "the original part" to represent his current BIM model as a data exchange standard, runs the prototype "CVP" from the Revit application, fills the login information, and selects the operation mode as "sender". Figure 6.26 shows the first operation screen (Registration and Selection) for accessing permission and operation mode. The "Select mode of Action" box is locked until the architect finishes with the right to access the prototype.

The steps of running the CVP are explained based on the sequence of events that followed in Section 6.3 and in Figure 6.24.

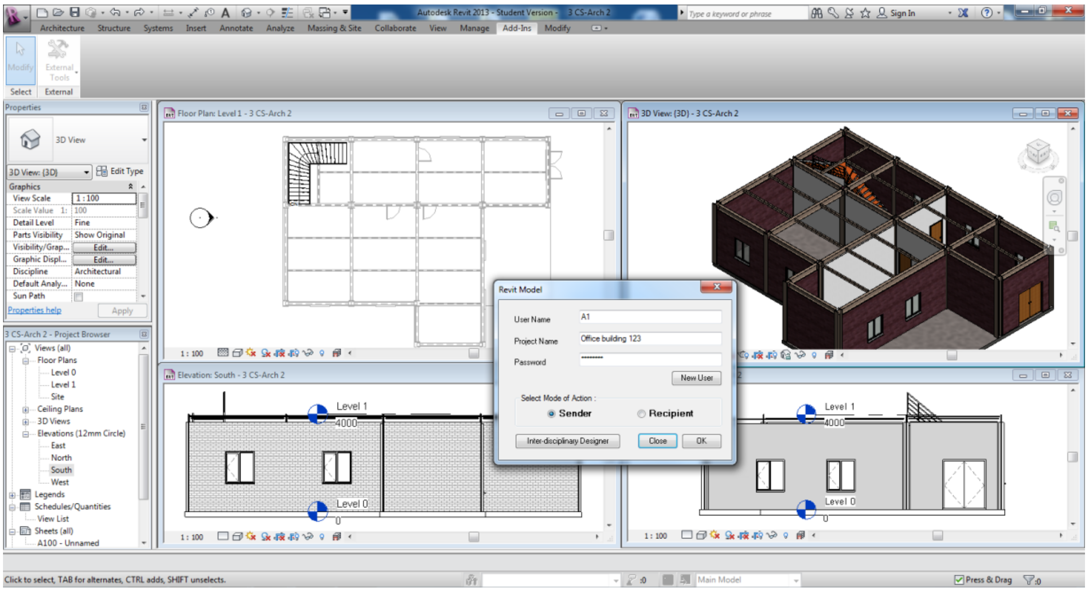


Figure 6.26 Supply of project information and selection of operation mode.

- **Modelling/ Sender (step a)**

The next window in the sequence of events is the "Sender" Window. It includes three tab pages. The first tab page is "Modelling". The sender within this page selects the information that is required to be extracted from the files (Figure 6.27). This includes element types and feature components. In this example, the architect (A_1) selects two building elements (beams and columns) with all available features. The sender then selects his IFC model to extract all the information related to beams and columns with their respective features. The operation involves calling up Windows Open-dialog to select the IFC model. Once this is complete, the process to find one or more of the IFC models in the cloud(s) is done automatically. Since beams and columns are shared elements between the disciplines, then there is a need to call the model version (M_1) from the global cloud. As a result, all the available beams and columns in the current and global cloud are listed in tables on this page.

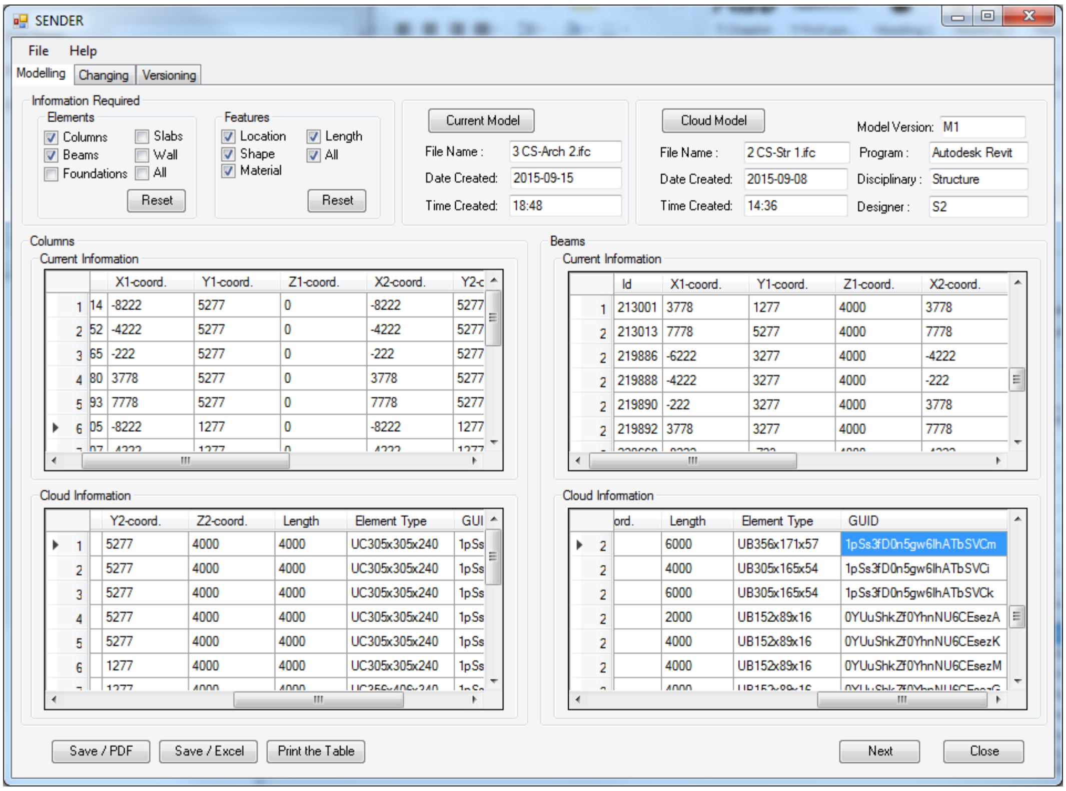


Figure 6.27 Display the required information in tables/ Sender

- **Versioning/ Sender (step b)**

Figure 6.28 shows the sample output on the "Changing" Page for the comparison of the two tables (the current and cloud tables) for each element (beams and columns). The comparison covers tracking every single change in the values of the features. For each element type, all added elements and new values of modified elements are represented in a table of "available changes" and all deleted elements and old values of modified elements are represented in another table" of "missing changes". Therefore, two tables for the beams and the same for the columns are created.

In the present example, the current changes include two additions, two modifications and four deletions for the beams and one addition, one modification and one deletion for the columns. Figure 6.29 illustrated these changes.

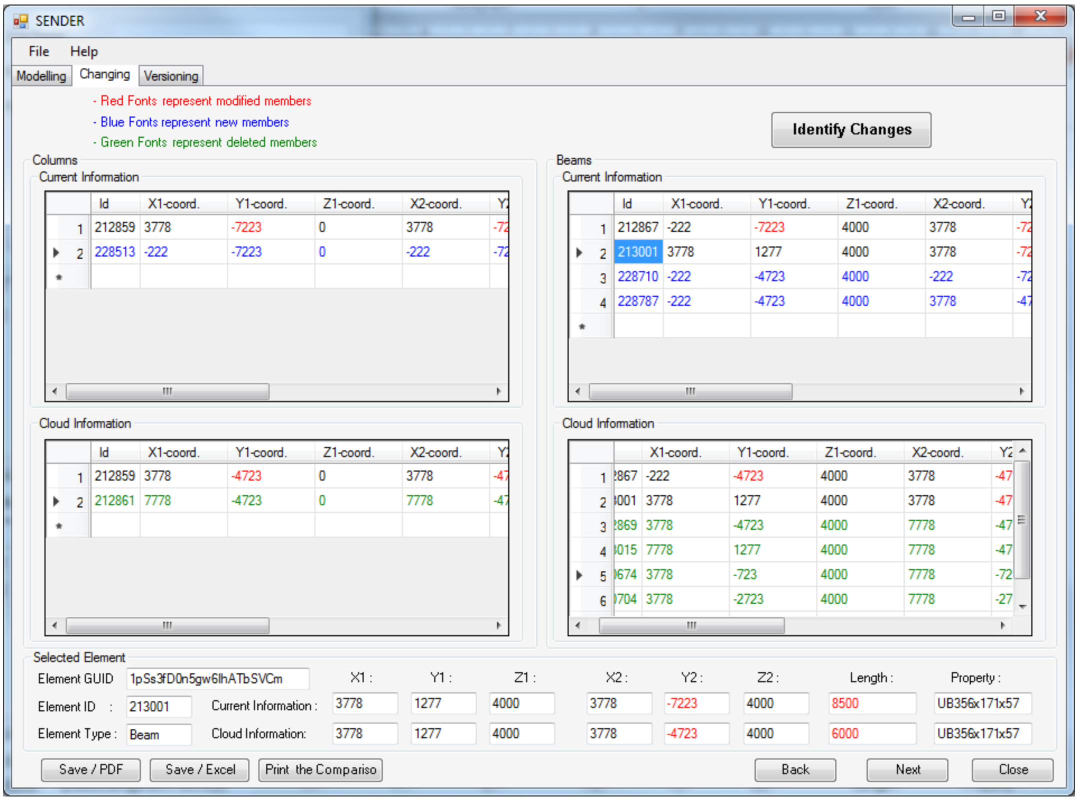


Figure 6.28 Display the changed information in tables

As evident from the tables, different colours are used to highlight the changed values. These colours are red for modification, blue for addition, and green for deletion. The final event at this juncture, the option of selecting an element from any table is provided to further illustrate the changed values in the features. For instance, beam with GUID "1pSs3fD0n5gw6lhATbSVCm" is the modified element in the current model that has been illustrated in the figure.

The screenshot for creating the architect (A_1) new model version is given in the "Versioning" Page (Figure 6.30). Since the changes in the current architectural model are for beams and columns, only a new model version in the global cloud is created. A message is displayed at the top of the page clarifying this information.

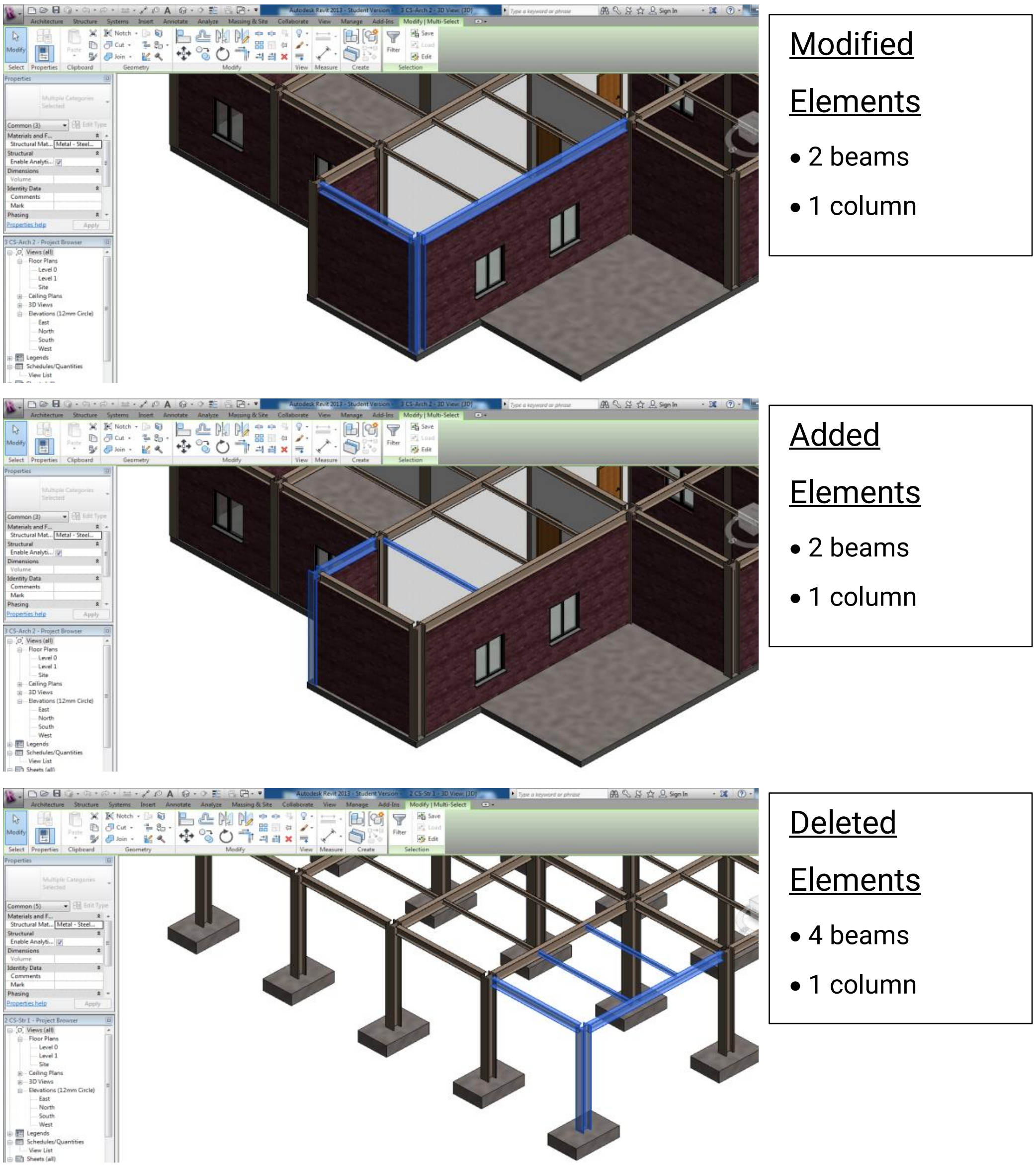


Figure 6.29 The changed elements in model version "M_2".

The identified changes for the beams and columns in the tables are translated into new entity instances based on the proposed extension to the IFC standard (section 5.2.1). The previous entity instances in the global cloud, which represent the versioning information in the “extended part” of the model version file “M_1”, are linked with the new entity instances for the current changes to create a new “extended part”. This part is added to the “original part” of the current IFC file of the architect (A_1) to generate a new model version “M_2”.

The last action on the sender side (A_1) is to send the global model version "M_2" to the local cloud to get the final approval from the architectural team (step 13). Since the sender, in this case, is the architectural team manager, he has the authorization to send the new model to the global cloud. Therefore, the architect (A_1) sends the new global model version (M_2) to the global shared folder and removes the current model version (M_1). Simultaneously, notification messages are then sent to the other designers (A_2, S_1, and S_2) about a new model version in the global cloud (step 14).

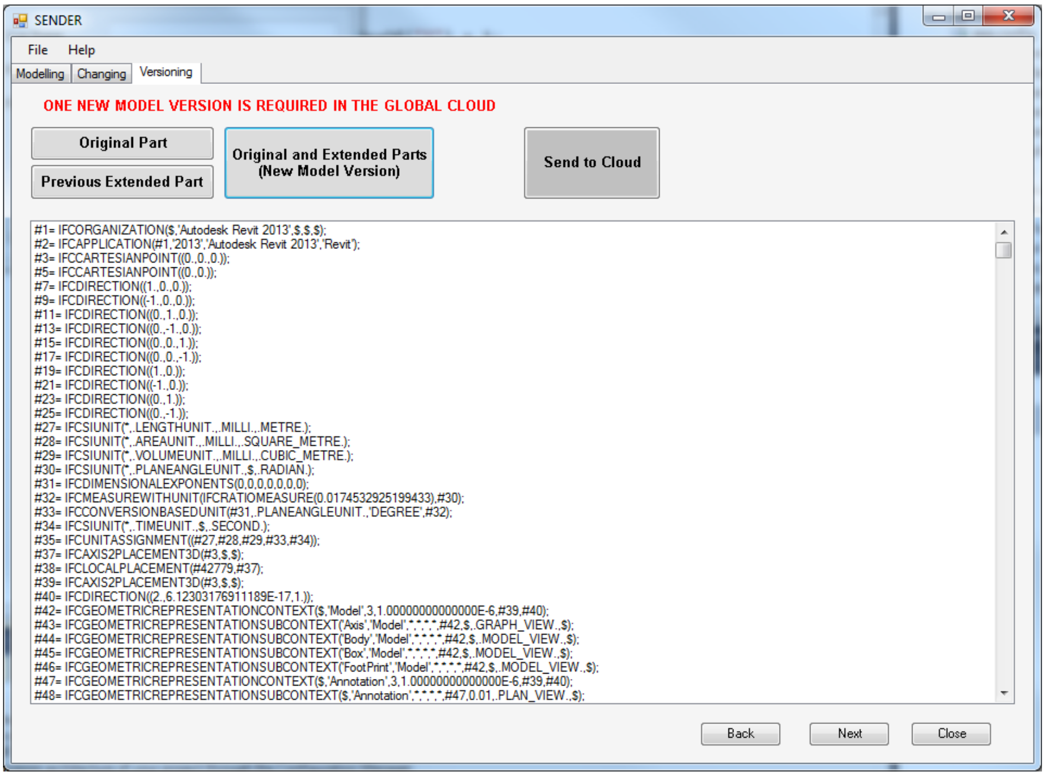


Figure 6.30 Create new model version "M_2"

- **Sharing (step c)**

A dropbox service is used to provide cloud storage for the latest model versions. Each disciplinary team has two shared folders (local and global folders) to save the two new model versions (local and global versions) respectively. For the current example, the model version "M_2" is in the global cloud and the model version "M_{S0}" is in the local cloud of the structural team. (S_1 and S_2) can access the both models while (A_1 and A_2) can access only the model version "M_2".

For the example at (step 9), when the structural designer (S_2) send the model version (M_1) temporary to the local cloud of the structural team and notified email sent to the structural manager (S_1). After getting the final approval about the model, the designer (S_1) then login the prototype, and selects the "inter-disciplinary designer" button (Figure 6.31). In the next window, the designer needed to accept transferring the new model version (M_1). The new model then sent from the local to the global cloud and the old model (M_0) is deleted and replaced with the new model version (M_1), which will be the only file in the global cloud. At the same time, a notification message is sent to all the participants in the project.

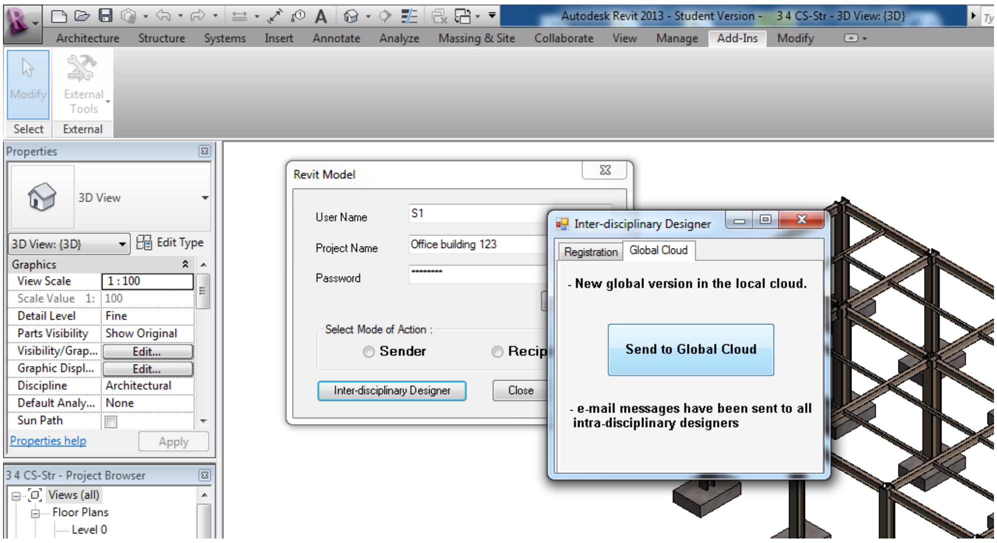


Figure 6.31 Transfer the new model version to the global cloud.

- ## Modelling/ Recipient (step d)

The process is moved to the recipient designers to read the new model version and display it in their model. This process has happened twice at two different times. The first is when the intra-disciplinary designers of the same architectural team (A_2) access the CVP to review the new model version and give their final decision about the changes before sending the file to the global cloud. The second is when the intra-disciplinary designers of the structural team (S_1 and S_2) access the CVP. The process for the both scenarios is the same, so only the second scenario is discussed in detail. After receiving the designers (S_1 and S_2) reminder messages from (A_1) about the new model (M_2) in the global cloud (step 14), each recipient needs to access the prototype through his Revit Model. The structural designer (S_1) is taken as an example in this example for dealing with the new model version (M_2). Figure 6.32 shows the current BIM model for the structural designer (S_1) with highlights of the changes since sharing the last version (M_1).

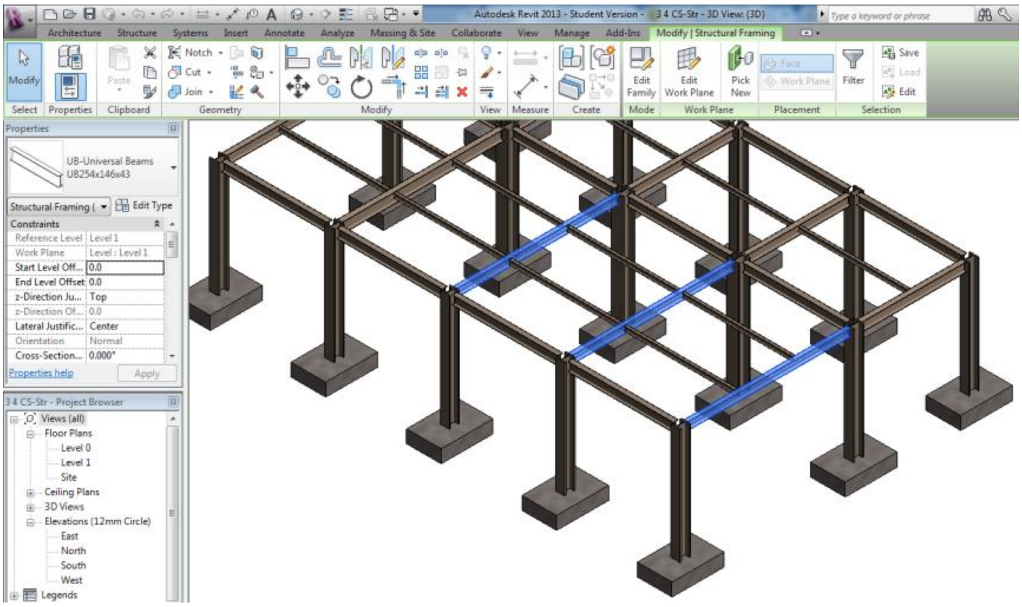


Figure 6.32 Changed elements in the current structural model.

On the recipient’s side, the steps of calling up the program, logging in and identifying the required information, are the same as in the sender side. The next window in the sequence of events is the “Recipient” Window. It includes three tab pages. The first tab page is “Modelling”. The process then continues with calling the new model version (M_2) from the cloud, and reading the model, regenerating the information of the previous model version (M_1) from the versioning information of (M_2), calling and reading the current model of the structural designer (S_2), listing all the general information, and displaying all the information about the beams and columns from these models in tables. All these processes are done automatically. Figure 6.33 shows a snapshot on the “Modelling” page for displaying the information of the new and old model in the cloud and the current model in tables.

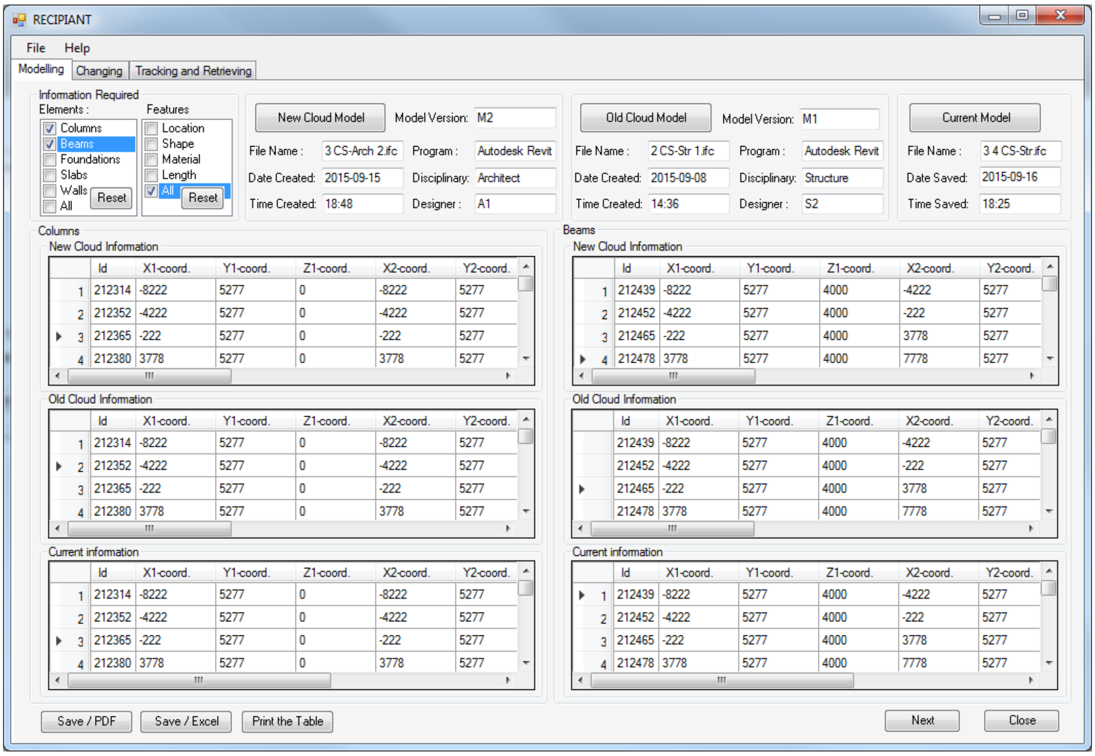


Figure 6.33 Display the required information in tables/Recipient

- **Tracking (step e)**

The change information is implicitly with the “extended part” of the shared IFC version. This information is used to extract the latest changes of the new model version (M_2). The comparison among the tables’ contents of the last section is done to track all current changes based on the latest changes of (M_2).

A set of comparative tables have been provided to log all the changed information. Figure 6.34 shows a snapshot on the “Changing” page. Due to the synchronous design, the structural designer (S_1) has changed some of information (two beams) from his model that has been changed at the same time in the new model (M_2). These shared changes, in addition to the current changes, have been highlighted in a different colour in the tables to make them more visible to the recipient.

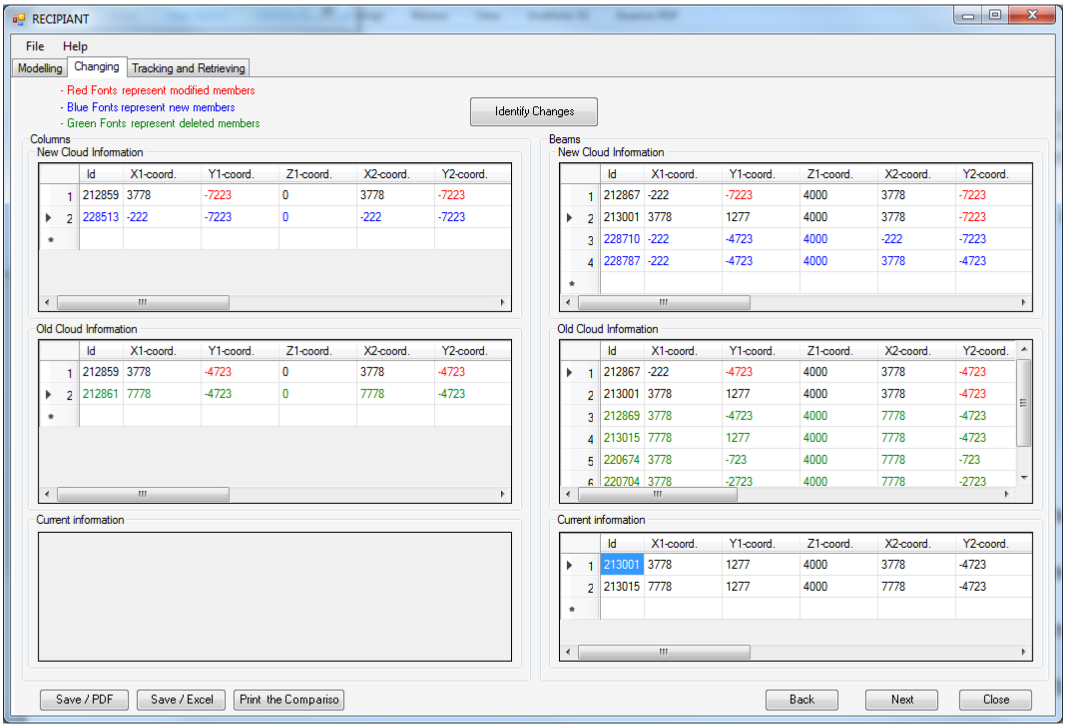


Figure 6.34 Display the changed information in tables/ Recipient

The next screen tab is “Tracking and Retrieving” page (Figure 6.35). Three spaces for the additions, modifications, and deletions elements are presented to display the whole or affected changes. The designer can select any element ID from the tables to display all features values beneath in a way that he can compare the same feature values in the new and old cloud models and the current model easily. One of the two shared beams has been displayed (beam 1pSs3fD0n5gw6IhATbSVCk) in the figure. This beam has been deleted in the new model version while it has been modified in the current model.

Additionally, graphical representation of the design change in the model is essential for engineering design systems to enable the designers to understand the changes visually. The added elements can then be drawn and the modified and deleted elements can be highlighted individually or all at once in the BIM model of designer (S_1).

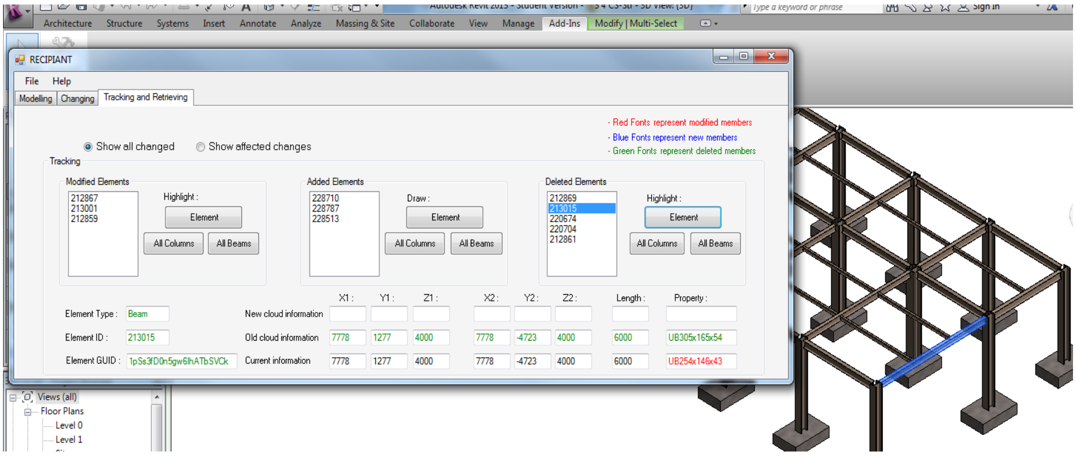


Figure 6.35 Track the changes numerically and graphically

- ## Retrieving (step f)

In the same screen page "Tracking and Retrieving", the structural designer (S_1) can display the changed history at the feature, element, and model levels. All the versions for an element or for a feature of an element can be displayed from its creation until the current state. In addition, the prototype can track and filter the change history and retrieve any earlier model version of the design. Figure 6.36 shows a snapshot of the "Tracking and Retrieving" page. The structural designer (S_1) presents the history information of the evolution of the second shared beam (1pSs3fD0n5gw6IhATbSVCm). This element is generated in (M_0) and then has been modified twice in (M_1 and M_2).

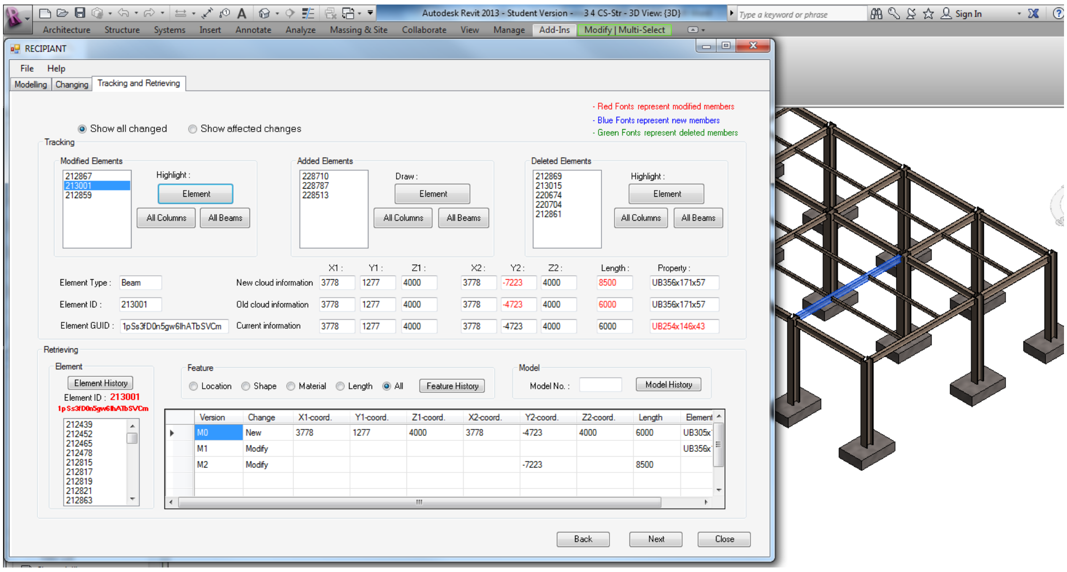


Figure 6.36 Retrieve the change history

At the end, the operations presented in the previous steps are repeated regularly to generate more local and/or global model versions in the cloud by both designers' teams.

6.5 Summary

The implementation of the collaboration versioning framework in the form of a prototype was discussed in this chapter to verify the efficacy of the versioning concept in an extended IFC model. It covered description of how the framework transforms to the implementation components. Moreover, the prototype is capable of not only implementing the extended IFC but also enhancing and supporting the completely collaborative design process. The developed prototype was generated using the Microsoft C# programming language and integrated into Autodesk Revit through using the API platform. The representations used in the implementation entailed a set of flowcharts provided by the different actors to describe the sequence of the collaborative versioning events (modelling, versioning, sharing, tracking, and retrieving).

This chapter also presented an example to demonstrate and validate the use of the proposed prototype system in a typical design activity. It explored a real design scenario involving multi-disciplinary designers and multiple-model versions. For further substantiation through an evaluation process, the intention of this phase of the research is to apply the collaboration versioning prototype on an example as an evidence of its effectiveness. The evaluation phase is presented in the next chapter.

Chapter 7

Industrial Feedback

7.1 Introduction

The two preceding chapters discussed the implementation stage through extending the IFC standard and developing prototype system. The validations of the versioning concept in the IFC-STEP file and of the Collaboration-Versioning Prototype based on case studies were also illustrated in these chapters. This chapter assesses the versioning system from the user's perspective.

The chapter starts with discussing the objectives and methodology of getting feedback (Section 7.2), followed by illustrating the procedure of using a questionnaire (Section 7.3). Design of the questionnaire and analysis of the results are then presented (Section 7.4), and finally some conclusions are drawn (Section 7.5).

7.2 Feedback Objectives and Methodology

feedback is delivering information and giving of meaning to the prediction of actual impacts of a proposal or result (Rossi et al., 2003). The objectives of the feedback process in this research are to assess extending the IFC standard to manage the versioning information and assess developing the Collaboration Versioning Prototype (CVP). It is intended that getting comment from the feedbackers will provide the following information:

- The appropriateness of extending the IFC standard to deal with the change information.
- The applicability of using the Collaboration Versioning Prototype within the design process and whether it offers a step change from the current methods.
- The suitability of versioning the current and history of changes for different BIM models.
- The suitability of saving the versioning model on a shared storage as a central data model to be used by all relevant disciplines.

Getting feedback offers a way to determine whether an initiative has been worthwhile in terms of delivering what was intended and expected (ICAP, 2012). Feedback methods may contain questionnaires, interviews, focus groups, and observation (Schensul et al., 1999). Selecting the right method involves many factors. Some methods are better for gathering quantitative data, others for qualitative data. Some are better for particular audiences than others. Some methods gather richer and deeper data than others do (CDC, 2011).

Feedback has been applied as an aid for software development during the last decade. Software can be assessed with respect to different aspects, for example, functionality, reliability, usability, suitability, maintainability, and portability. These aspects have gained particular importance with the increasing use of interactive software (Gediga et al., 2002).

Different approaches can be adopted to assess the performance of different factors of the research. This research adopted a questionnaire survey as a key tool in this methodology. A questionnaire is a set of questions for gathering information from individuals. It can be administered by mail, telephone, using face-to-face interviews, as handouts, or electronically (i.e., by email or through Web-based questionnaires) (McLafferty, 2003). A questionnaire is vital to gather information and to get feedback. It also gathers opinions from respondents, in a form that can be analysed. This method has been found to be appropriate to this research since it is simple and effective for collecting information from a large number of people (Chae, 2015).

7.3 Feedback Procedure

The number and the type of the participants in the questionnaire are discussed here. Also, the procedure followed for collecting information and obtaining feedback are also included in this section.

7.3.1 The selection of participants

The research objectives were presented to a group of building designers to gauge their response through a questionnaire. The selection of participants can depend on several factors, such as “Efficiency, Experience, and Truthfulness” (Struck 2012). The number of participants recommended to be involved in the questionnaire varies. Holzinger (2005) stated that inspection methods require 1-5 participants, whereas test methods need 4-30 participants. It was decided that ten participants are sufficient in order to confirm the validity of extending the IFC standard and the collaboration versioning system. Seven of them are working in the AEC industry and the other three are academics in building design. The intention is to capture the opinions of different AEC disciplines into this feedback process. Therefore, the ten feedbackers are distributed as three architects, five structural designers, one mechanical designer, and one electrical designer.

The participants can also be categorized as (innovators, early adopters, and conservative) (Hopfe et al., 2005). Table 7.1 displays the definition of these categories in terms of BIM use. Consequently, the ten participants of the survey are categorized as one innovator, five early adopters, and four conservatives.

Table 7.1 Different category of BIM users

Category	BIM adoption	participants
Innovator	Develops BIM tools	1 Str
Early Adopter	Uses BIM regularly	2 Arch, 2 Str and 1 Mech
Conservative	Uses BIM occasionally	1 Arch, 2 Str and 1 Elec

Arch: Architect, STR: Structural designers, Mech: Mechanical designer, Elec: Electrical designer

7.3.2 Information Collection

A series of questions were prepared for gathering information and obtaining feedback from both individuals and groups. A presentation of the application and its operation was given by the author to demonstrate different aspects of the system. The presentation was carried out as clearly as possible and participants were encouraged to be critical in their responses.

The following procedure for collecting the information was followed to ensure that the feedback fully understood the objective of the work:

- Presentation: A presentation was given by the author to introduce the theoretical aspects of the system. It covers aspects related to collaboration, multi-disciplinary designers, different BIM models, change management, versioning, extend IFC, change history, and cloud storage. Slides of the presentation are provided in Appendix 1.
- Case Study: A demonstration was given by the author to introduce the implementation aspects of the system. The demonstration illustrated a typical case study scenario between the sender and the recipient (same as that shown in chapter 6). The representations used in the implementation described the sequence of the collaborative versioning events (modelling, versioning, sharing, tracking, and retrieving). Two videos were prepared for this Case Study so that the feedbackers could see how the system works.
- Questionnaires: A series of questions were given to the feedbackers to assess the system. The questionnaire contains twenty-six questions classified into five main groups (general impression, design process, design changes, versioning, and central shared storage). The questionnaire is provided in Appendix 2.

The above three procedures are provided in a cloud storage service (dropbox) as a shared folder. Each feedbackers has the choice of either filling the questionnaire immediately or later. If done later, feedbackers can send the filled questionnaire to the shared folder via the link provided. With the shared folder, the feedbackers can go through the slides and videos again at their convenience before answering the questions.

7.4 Design and Analysis of the Questionnaire

Questionnaires must be designed carefully so that answers should reflect the required information. Design and analysis of the questionnaire are discussed in this section.

7.4.1 Design of the Questionnaire

Twenty-six questions are prepared for this study (twenty closed-ended and six open-ended questions). The intention of the closed-ended questions is to limit the answers of the feedbackers to concrete options provided on the questionnaire (using five-level Likert scale). The intention is to allow the use statistical analysis. On the other hand, the open-ended questions allow the free expression of opinions that would have been difficult to capture with the previous method. It provides rich qualitative information for the researcher with an opportunity to gain insight on all the opinions on a topic they are not familiar with. This method usually provides the answers to what, how, which or why type questions (Jacko, 2012).

As mentioned above, the questionnaire is classified into five main groups (general impression, design process, design changes, versioning, and central shared storage). Each group includes four closed-ended questions and one open-ended question as well as one general question at the end. The questions of the closed-ended method are presented as numbers (such as, Q_1) and the questions of the open-ended method are presented as letters (such as, Q_A) (see Appendix 2).

7.4.2 Analysis of the Questionnaire

All the ten feedbackers answered the twenty six questions. After analyzing the responses of the feedbackers, the results generally indicate that they had positive opinions about extending the IFC and the collaboration-versioning system. The most responses for the twenty closed-ended questions on general impression, design process, changes in BIMs, versioning, and central shared storage were either “agree” or “strongly agree”. The results for the closed-ended questions (Q_1 to Q_{20}) are given in Figure 7.1 to 7.5. The responses of the feedbackers are further discussed.

- General Impression (Figure 7.1): There is wide impression that the collaboration- versioning system and the idea of extending IFC to deal with design changes will improve the design process. Moreover, there is a good agreement that the system had positive impact multi-disciplinary designers to collaborate earlier.
- Design Process (Figure 7.2): most of the feedbackers agree that the system displayed a good degree of flexibility in operation. It can help reduce re-design possibilities and decrease design process time and cost. There is a general agreement that the system improves the collaboration among different designers. One feedbacker did not support the idea of using local and global model version and use them in different shared clouds.
- Changes in BIMs (Figure 7.3): Most of the feedbackers share the opinion that extending IFC to deal with design changes is very useful. Also, most of them were satisfied with the process of identifying design changes numerically and graphically.
- Versioning (Figure 7.4): all feedbackers agree or strongly agree that versioning at the element and feature levels improve traceability of changes. Most of them are supportive of adding new entities into IFC schema to create versioning of the changes for different BIM models. One feedbacker preferred store different model versions to return to earlier stages of design.

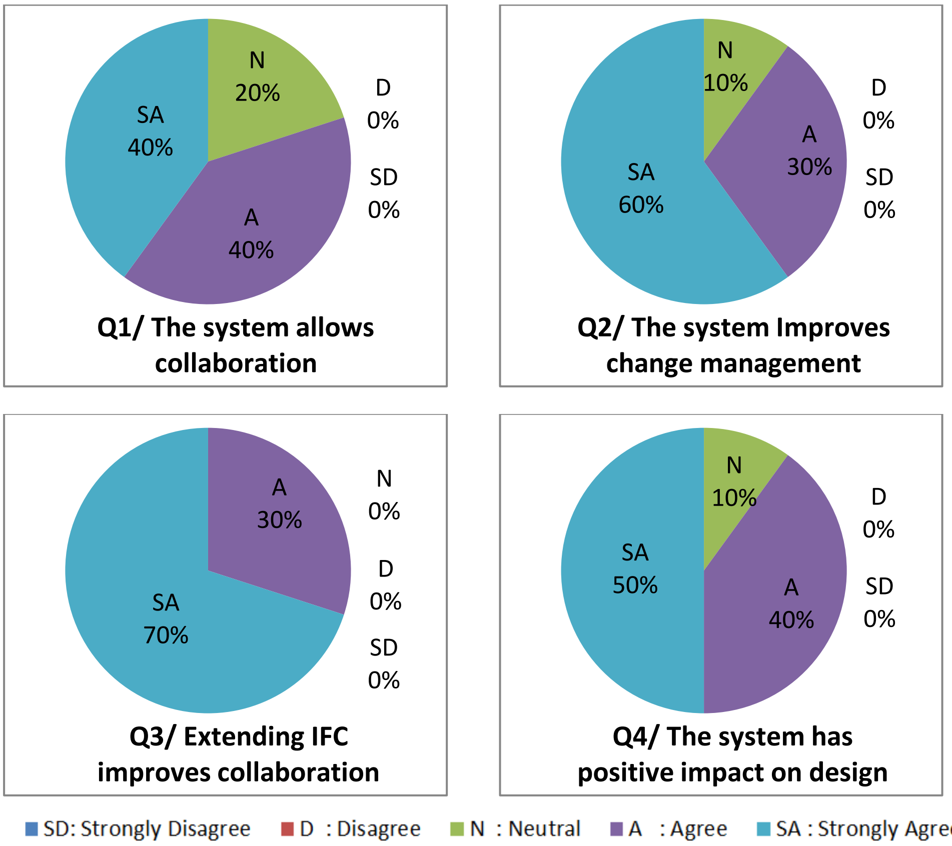


Figure 7.1 Response from feedbackers for “General impression” questions

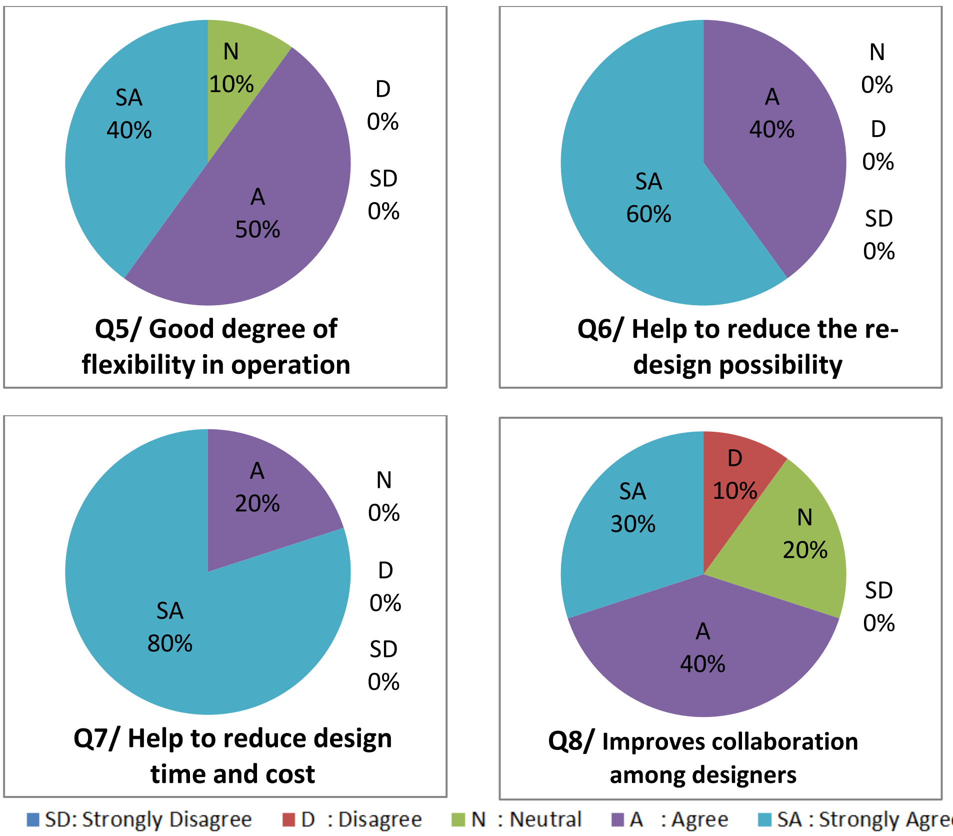


Figure 7.2 Response from feedbackers for “Design Process” questions

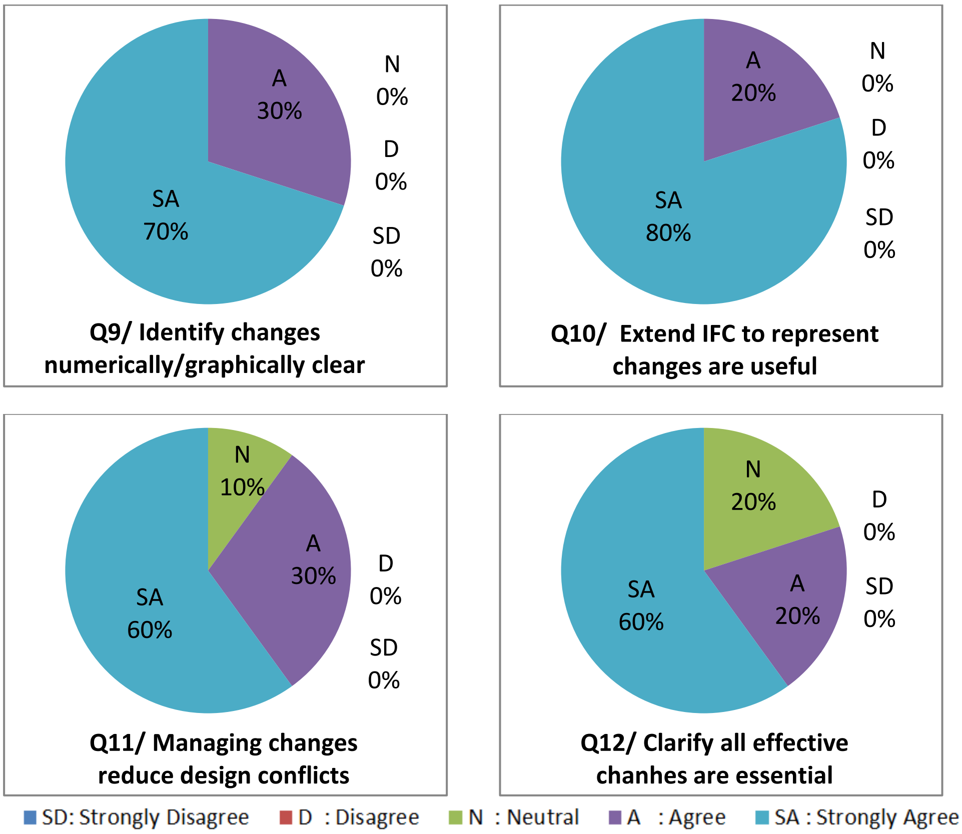


Figure 7.3 Response from feedbackers for “Changes in BIMs” questions

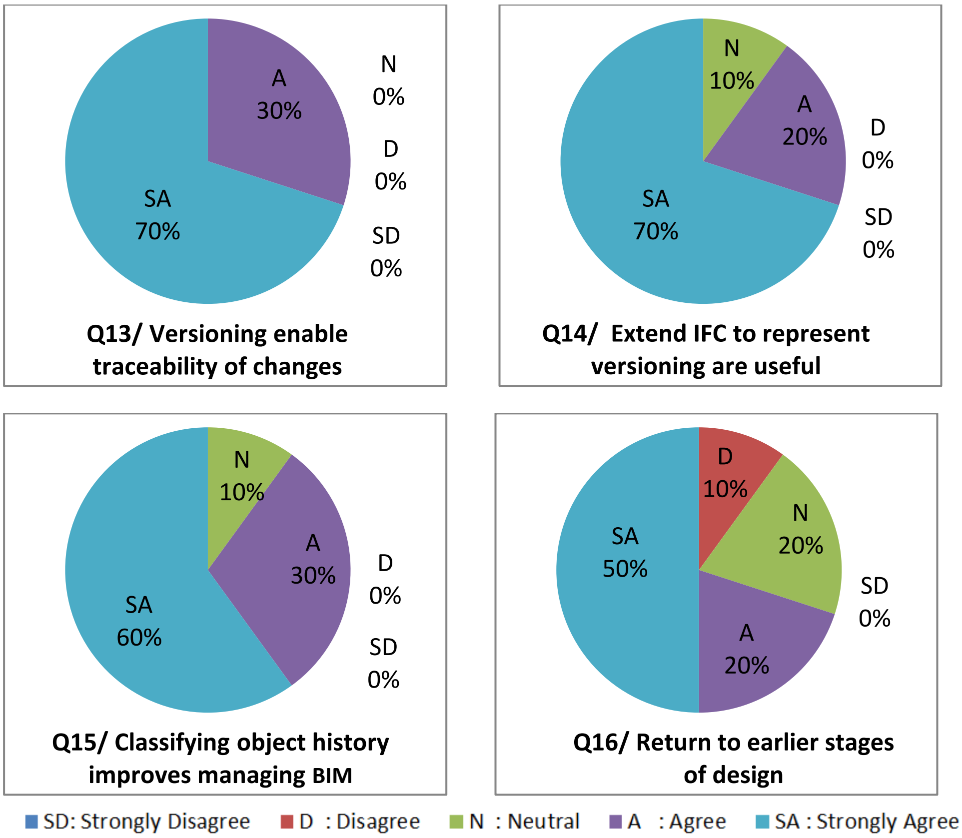


Figure 7.4 Response from feedbackers for “Versioning” questions

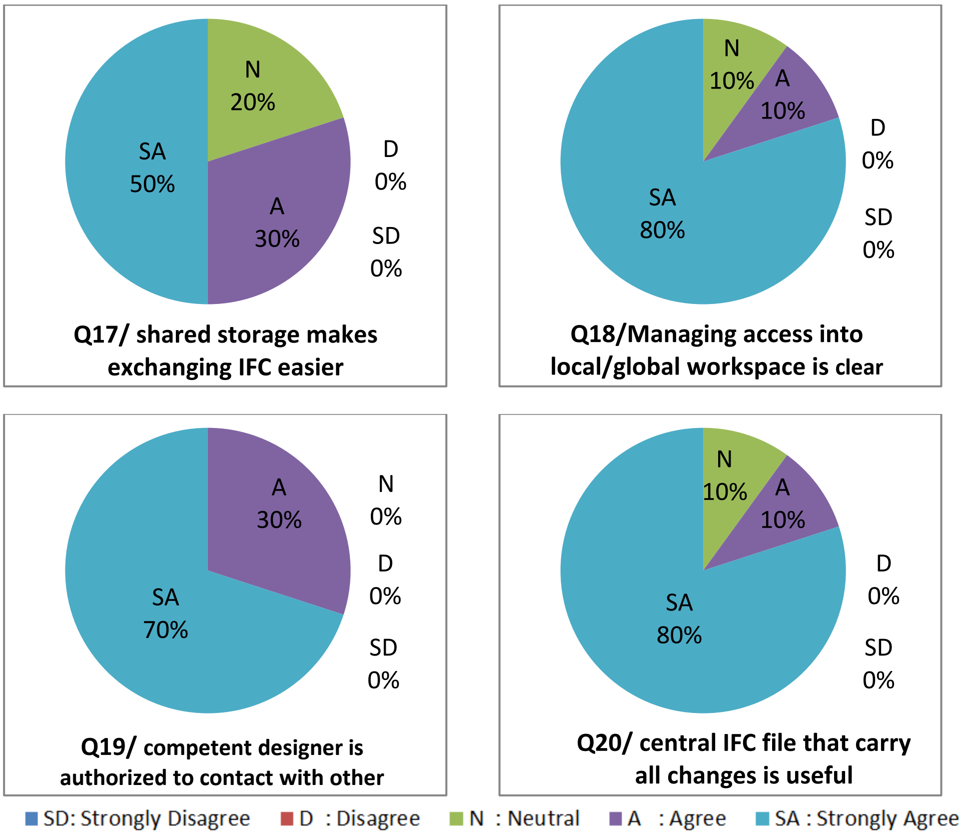


Figure 7.5 Response from feedbackers for “Central Shared Storage”

- Central Shared Storage (Figure 7.5): Most of the feedbacker supports strongly the idea of using local and global workspaces among intra- and inter-disciplinary designers to share the new model version. Furthermore, using cloud service to share the updated file in a manageable way was desirable by most of them.

The responses on the open-ended questions provided a great deal of inferences for getting feedback the research work more than on the closed-ended questions. The corresponding responses from the interviewees for the six open-ended questions (Q_A to Q_F) are discussed next:

Q_A. Which parts of the system you found that affect positively the collaboration process? And why?

The intention from this question is to gather information on the stages that the feedbackers found more useful in the collaborative processes associated with changes in BIM between the sender and the recipients. Five choices were given to the feedbackers (modelling, versioning, sharing, tracking, and retrieving). This will help improvement of the prototype and recommendation for further implementation refinement. Below are the impressions of the feedbackers to the selected stage:

- Modelling: The feedbackers were impressed with the idea that the sender extracts the latest change information and that the process of comparing the new with the previous models has been done at the sender side.
- Versioning: all the feedbackers strongly agree to extend the IFC standard to deal with the change information and to versioning this information in such a way that a single IFC file can store the current with all change information.
- Sharing: the idea of two workspaces for the intra- and inter- disciplinary teams admired the feedbackers that the specific changes are separate from the shared changes.
- Tracking: They were also impressed with displaying the affected changes graphically on the BIM model of the recipient. They mentioned that this is really useful in saving time in that designers do not have to manually display the changes.
- Retrieving: the ability of the prototype to retrieve any old information from the current model version is very important that any designer can address the evolution of any feature or element at any stage of the model.

The feedbackers could choose more than one stage for the sequence of the collaborative versioning events. The numbers of feedbackers that support each event are presented in Figure 7.6.

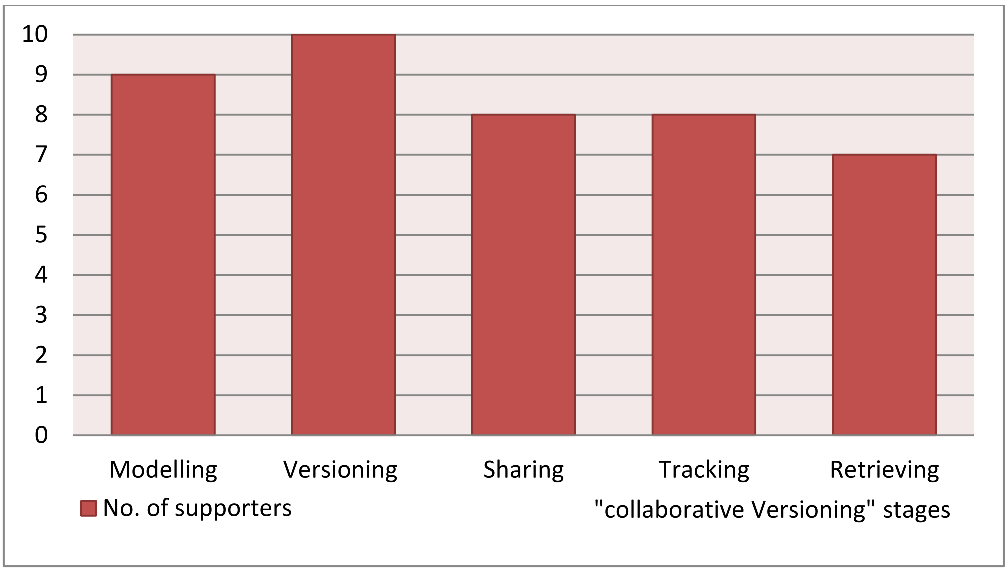


Figure 7.6 Response from feedbackers for open-ended question (QA)

Q_B. What are the obstacles to the adoption of such system in the design process?

There was a common feeling among the feedbackers that the introduction of any new idea or software needs to take into consideration cultural and technological obstacles. The attitude, feeling, interest, and intent of the designers towards accepting the changes brought by new technology are one of these obstacles. Awareness of the existence of such program in the engineering field among practitioners will take time to encourage its use. Some feedbackers believe that even with available some Software (like solibri model checker), which are dealing with the change information, the CVP may become acceptable for used in the design process because of the new and different concepts that characterize the program.

Q_C. What do you think of the advantage that the sender can illustrate all the changes in the new model to all other disciplines?

The feedbackers were impressed with the idea that instead of "n" numbers of designers do the comparison process between the new with the previous models and track the changes; only one designer who issues the new model version can do this process and clarify all the effective changes. Some feedbackers from the industry went further and mentioned that this new approach will require re-engineering the design process and this will reduce the time and efforts spend and enhance the collaboration capabilities.

Q_D. Do you support the idea that versioning all changes in IFC can improve the design process?

The feedbackers supported extending the IFC standard so that it not only deals with storing the current state of the model but also can be extended to manage and version the change information to handle even all previous changes. Some encouraged that this contribution does not remain just as a proposal thesis but also could be used to formally accept in the IFC standard.

Q_E. Do you agree with the idea of having an IFC file stored in the cloud that carrying the current information with all change history?

The feedbackers considered the idea of having a unique model version every time in the cloud is a new and efficient approach for the change management process. Instead of dealing with "n" numbers of IFC files ("n" model versions), only one developed IFC file (the latest model version) can be equivalent to them. Moreover, the fact that this unique file is a centrally shared among the related disciplinary team has received recommendation by the feedbackers. Some of the feedbackers argued that this will help enable team members to work more effectively as a single unit.

$\mathbf{Q_F}$. Any additional suggestions and further comments:

Aspects on expansion and improvement of the versioning concept and the prototype were mentioned as desired additional suggestions. These suggestions are presented below:

- Increasing the library of the building elements and the features in the CVP to cover more information for tracking the changes during the design process.

  The main building elements (beams, and columns) and their features (geometric shapes, spatial location, and material information) are implemented in CVP. Further research is needed to deal with the IFC schema of the other elements.

- Include further disciplinary teams in the design process in the CVP. Such as the MEP, client, contractor.

  The case study that had been shows to the feedbacker was between the architectural and structural teams. The CVP can deal with all the disciplinary teams involved in the project.

- Separate the current and the versioning information into two IFC files to avoid dealing with one big file.

  The size of the extended IFC file is not that big compared to the traditional file and also there are many direct links between the current and the versioning information. But this suggestion is significant and can be taken into consideration in future expansion.

- Interaction and communication among designers to deal with the design changes.

  The communication among the designers is another big area of research that need to study the response of each designer on each design change and build a communication among them to accept or reject the change order, which is beyond the scope of this research.

- Using the other object-oriented BIM platform (such as ArchiCAD) and not only Autodesk Revit to link and execute the CVP.
  From the research perspective, it is possible to link the CVP to any BIM application, Autodesk Revit was used because it is one of the most commonly used BIM applications among designers.

The feedbackers were impressed with the level of automation in the extraction of information from the building model and the generation of the extended IFC file. They added that it is an interesting field of research and marketing of the prototype is worth considering and be adopted in the design process. The feedbackers confirmed that the extended IFC and the CVP will improve the collaboration among multi-disciplinary designers.

7.5 Summary

This chapter described getting feedback of the proposed extension of the IFC standard to deal with the versioning concept and of the development of the prototype system. A questionnaire with Case Study videos and PowerPoint slides were prepared and demonstrated to ten feedbackers. The results showed that this is not only affirmative but also that it can support the collaboration among multi-disciplinary designers through managing changes from the beginning of the project. Future improvements were also suggested for the current work.

Chapter 8

Conclusion and Recommendations

8.1 Introduction

This chapter gives a recap of the research aim and objectives and a summary of how they have been realised in the course of this research (Section 8.2). It also summarizes the research questions and findings (Section 8.3), contributions (Section 8.4), recommendation for further work (Section 8.5), and list of the research dissemination (Section 8.6).

8.2 Research Summary

Changes in AEC projects are common at any stage of a project, by different designers, and on different models. Even with carefully controlled design process, it is inevitably prone to numerous changes and revisited decisions at various times of the project lifecycle. Poor management of the design changes has unforeseen side effects on time, cost, and quality of the project. As discussed in chapter 1, 2, and 3, there are several challenges and limitation related to managing the different changes in the IFC standard and in the design process. The following summarizes these limitations:

- Limitations in the IFC standard:
  - It only provides an indication of the change type of the elements without clarifying the change details. This indication approach is not active in the most BIM applications.
  - It does not deal with the change information of the element features.
  - Versioning concept is missing in the data schema architecture of the IFC.
  - It does not store the previous changes of the model.

- Limitations in the current design process:
  - Collaboration concept is mostly limited to sharing the new release of the BIM model among the disciplines without identifying the changed information.
  - Versioning concept is done at the model level only without taking into consideration of the versioning the building elements or their features.
  - Each disciplinary designer needs to compare the new and the old model versions to identify the changes.
  - Much of the changed information in the new model version could be irrelevant to specific disciplinary.
  - All model versions have to be archived in an individual or a shared place for a possible re-inspection in the future.
  - To find specific inform, designers are forced to review all of the model versions (files).

- Some proprietary BIM applications can only find the design changes in the model file that is compatible with their own models (e.g. Revit).
- Some proprietary applications are specialized to find and check the design changes only through comparing two model files (e.g. Solibri Model Checker).

Based on the above observed limitations, the research aim was to develop a methodology to improve the collaborative design process by versioning the design changes of different BIM models based on an open data exchange standard. A good way to achieve this was through using and extending the IFC standard to process and version the design changes. The benefits of extending the IFC standard were used to improve the whole design process through creating prototype software that can implement the extension as well as deal with multi-disciplinary teams, different BIM models, and shared storage server. To achieve the overall aim of the research, the set of objectives and achievements are as follows:

- **Objective 1:** Investigate the state of the art and identify shortcoming in managing design changes in building information modeling (BIM) and data exchange standard (IFC).

The literature review was conducted and presented in two chapters. Chapter 2 outlined the design stages and clarified how collaboration is essential for the success of engineering projects. It continued with the review of BIM adoption in the design process and then provided an overview of possible changes on the building models and how they can be managed during the design process. Chapter 3 focused on reviewing interoperability in BIM and provided an overview of the IFC standard. Limitations of adoption of BIM in the design process (Chapter 2), and limitation of managing the design changes (Chapter 3) were identified and discussed. This helped in identifying research methodology associated with managing design changes through incorporating the versioning process in the IFC standard and developing new prototype software.

- **Objective 2:** Identify the requirements for modelling a collaborative design process using the versioning concept in the IFC standard.

The development process of a proposed methodology started with identifying the requirements that were used to guide designing the collaborative process. The requirements that are described in the first part of chapter 4 are in two aspects; the first aspect is concerned with extending the IFC to embed the versioning concept and the second aspect is for implementing a prototype to assess the use of the extended IFC. For the first aspect, Minimality, comprehensiveness, generality, and consistency were identified to be important. Centrality, scalability, visuality, historicity, manageability, and automaticity are the requirements for the second aspect.

- **Objective 3:** Propose a collaboration versioning framework to capture various factors influencing the versioning concept to enhance the collaborative design process.

The collaboration versioning framework was described in the second part of Chapter 4. The architecture of the extended IFC in the collaborative design process was outlined. Many components of the framework were discussed and many solutions of versioning the IFC were provided. Three types of affected information to the designers (affected, semi-affected, and unaffected information), two types of versioning among designers (global and local versions) and two types of sharing workspaces among multi-disciplinary designers (global and local clouds) have been suggested. Moreover, three gradual levels (model, element, and features) for versioning the changes in the building information have been proposed. A new versioning graph was developed to study the change information only at different levels and a version history of the features of each element was proposed to review and retrieve any change of the element features.

- **Objective 4:** Implement and validate the versioning concept in the IFC standard through extending the IFC standard.

The implementation of the collaboration versioning framework for extending the IFC standard to cope with the versioning concept was fulfilled in Chapter 5. The versioning factors (version number, ownership, physical product, changing type) were examined and combined to generate two main entities in the IFC-EXPRESS schema to manage the design changes in the elements and features levels. The extension proposed of the IFC-EXPRESS schema was implemented in the IFC-STEP file to version and store the current and previous changes in the design information. This improves managing the model and allows tracking the changes that had been made by any designer. A simplified case study was presented to explain further the Implementation of the versioning concept and managing the design changes in the IFC-STEP file.

- **Objective 5:** Implement and validate a collaboration versioning prototype based on the proposed framework and the IFC extension.

The implementation of the collaboration versioning framework in the form of a prototype was discussed in chapter 6 to verify the efficacy of the versioning concept in an extended IFC standard. The Collaboration Versioning Prototype (CVP) was developed using Microsoft C# programming language and integrated and interfaced with Autodesk Revit .NET API. The prototype is capable of not only implementing the extended IFC, but also gives the opportunities of enhancing and supporting the whole collaborative design process. A case study was presented to validate the use of the prototype system in a typical design activity.

- **Objective 6:** Feedback on the effectiveness and efficiency of the proposed IFC extension and the prototype software.

Chapter 7 presented the feedback of the extended IFC and the developed prototype. A set of questionnaire with case study videos and power point slides were prepared and demonstrated to the feedbackers. The feedbackers supported extending the IFC standard to manage the changes and were impressed with the ease of use of the prototype. They were also indicated that extending the IFC is necessary to support the collaboration process among multidisciplinary designers through managing changes from the beginning of the project.

8.3 Research Questions and Findings

In order to improve the collaborative design process, main and sub-research questions were developed in chapter 1. The questions and their answers are explained below:

- **Main question: How can a BIM strategy be employed to manage design changes to better support multi-disciplinary collaborative design?**

The capabilities of the promised BIM strategy have been used to deal with different changes in different models. An IFC model, which is a data representation of BIM model, has been suggested to deal with managing design changes. The methods used to answer the main question were through:

- Version all design changes that occurred in different disciplinary models.

- Extend the capability of IFC standard to deal with the versioning concept.
- Develop a prototype to deal with the extended IFC standard.
- Take advantage of the benefits of integrating design changes into IFC standard to improve the whole design process.

The solutions above were further decomposed into two sub-questions.

- **Sub- question: How can the extension to an existing data exchange standard support the management of design changes?**

Integrating the versioning process and the use of IFC model has been suggested to deal with different design changes. This developed the capacity of data model to process dynamic data. The Integration is done in two parts:

- Develop the IFC EXPRESS schema to adopt the versioning approach through:
    - Analyse and use the existing entities that represent the elements, features, and ownerships to deal with different design changes.
    - Identify the factors (version number, ownership, physical product, changing type) in IFC that need to implement the versioning concept.
    - Extend the IFC schema through generating and adding two main new versioning entities to the schema to version and manage the design changes at element and feature levels.
- Implement the developed IFC EXPRESS schema in IFC-STEP file through:
    - Integrate the new versioning entities that represent the design changes into the existing data modelling.
    - Store the missing information (deleted and modified features of the earlier version) in the current model version and link them with the new entities.

- Link the available information (added and modified information of the latest model version) in the current model version with the new entities.
- Store the old changes (history information) in the earlier model versions and link them with the current changes.
- Validate versioning the IFC in a simple design activity.

- **Sub- question: How can the proposed extension be verified and validated?**

The feasibility and usefulness of the versioning concept in the extended IFC model has been verified and validated through:

- Implementing the extension as a collaboration versioning prototype (CVP) that can be used for managing different changes. This was done through:
  - Use Microsoft C# programming language to develop the prototype.
  - Integrate the developed prototype with Autodesk Revit.
  - Link the developed prototype with on-line storage “Dropbox.

- Use of the collaboration versioning prototype to improve the whole design process through:
  - Allow collaboration between multi- disciplinary teams through local and global models.
  - Manage the access permission for each inter- and intra-disciplinary designer.
  - Manage the shared and specific information based on the disciplinary teams.
  - Store and share the changed information on a server.
  - Validate the prototype system in typical design activities.

The key findings of this research are summarized below:

- Versioning the changes of IFC models can be incorporated within it for improving the collaborative design process.
- The new versioning entities that have been suggested within the IFC standard have significant effects in enhancing the collaboration among different disciplinary designers. The suggested entities followed the same structure of the inheritance hierarchy of the current IFC schema (2x4).
- Integrating the versioning concept into early design stages has greater influence on the total design process.
- The suggested approach provides consistency and robustness for the design process. In general, it reduces the time and efforts that spent in managing design changes and enhance the collaboration capabilities.
- Sharing a single unique shared file among different disciplinary designers is in an automated way.
- In the new IFC file, each element in the model holds its own change history since creation. This improves managing the model and allows tracking the changes that had been made by any designer.
- Recognizing the semantic of IFC contents in a way that designers can understand the meaning of the changes in the IFC model.
- Classifying the project information into three versioning levels (whole model information "M_V", element information "E_V", and feature information "F_V") improved managing the design changes.
- Classifying the project information based on disciplinary teams into two versioning types (global version "G_V" and local version "L_V") improved managing the design process.
- Classifying the storage workspaces based on disciplinary teams into two separate workspaces (global cloud and local cloud) improved managing the shared file.

- Compiling and view all changes in the information model have greater influence on visualizing impacts.
- Automating the processes of change management (comparing different models, finding different changes, clarifying the necessary and affected changes, creating the extended IFC file, sending alert e-mail message, sharing the new model version, deleting the old model version, showing the history information, and regenerating the old model versions) enhanced the collaboration among the participants.

The advantages of the developed program (CVP) over conventional programs (Revit software, Solibri Model Checker and BIMserver) in terms of managing changes during the design process are summarized in Table 8.1. The common characteristics are not listed in the table (e.g. compare files, find changes, view changes, etc.…).

Table 8.1 Comparison between the developed program (CVP) and some other applications

	CVP	Revit	SMC	BIMserver
The shared file has to be IFC.	√		√	√
Manage access permission based on disciplinary teams.	√	√		√
Use shared server to store the changes.	√	√		√
Use single file in the shared server.	√			
Separate the workspaces of disciplinary teams in the server.	√			
Store current changes implicitly in the model version.	√			
Store history changes implicitly in the model version.	√			
Versioning the IFC changes based on element and features.	√			
Classify project information based on the relevance to disciplines.	√			
Retrieve and view the old information.	√			

8.4 Research Contributions

The main contribution to knowledge of this research is the extension to the capability of the existing IFC standard to manage different design changes in different BIM models. It is the first work to incorporate versioning process, as a change management approach, in the IFC model, as data representation of BIM. The proposed extension provides advanced collaboration among different building professionals during the design process and beyond. These processes of managing the design changes covered an important gap in the IFC standard that can subsequently be embedded in the new release of the IFC standard.

Additionally, other research contributions to knowledge include the following:

- The research categorized the key requirements for managing a collaborative system. These requirements include extending the IFC standard to cope with the versioning concept and implementing prototype to cope with the extended IFC.
- The research provided a framework that could be used in the development of design systems to facilitate managing design changes and collaboration design processes.
- The research developed Collaborative Versioning Prototype (CVP) system for implementing the extended IFC and managing the whole design process.
- The extended IFC and the developed CVP improved the collaborative design through:
    - Developing a new way to version the design changes through IFC based on elements and features.

- Proposing a minimum set of new IFC entities to mimic the versioning requirements and manage the design changes.
- Designing new IFC entities that are generic to cover the whole elements and their features.
- Expanding the versioning domain to include the model, element, and feature information.
- Reducing the process of finding the design changes into one designer, who generates the new model version.
- Reducing the number of the model version files among each disciplinary team into a single file. This file is the latest model version that stores the current with the history information associated with different BIM models.
- Regenerating old model versions through using the history information within the latest (single) model version.
- Showing the evolution of elements and features through using the embedded history information within the IFC.
- Separating the global changes that affect all-disciplinary teams and the local changes that affect the intra-disciplinary team into two different versions.
- Providing meaningful to the changes to elements and features of IFC model through transferring them to understandable (semantic) information for the designers.

- Developed Change Evolution Graph to study the versioning of the changed information only at the feature level beside the model and element levels, and to retrieve the previous information.

- The developed IFC, for managing changes, and the developed system, for collaboration, can support Level 3 BIM maturity because:
  - BIM models are transferred into a manageable IFC format.
  - Managing the design changes are integrated into the IFC model.

- Data is integrated into one single project model, which is located online in a shared open server.
- The single shared project model can be accessed and modified by the actors involved based on permission roles.
- Full collaboration among multi-disciplinary designers and different BIM models is possible.

8.5 Recommendation for Future Work

This research proposed a fundamental approach for improving the collaborative design. Aspects of the work can be developed through further research. The most important of these are outlined below:

- Extend the IFC standard to tackle changed information in the structural analysis of buildings and other structures (e.g. dams, bridges, roads, etc.…).
- Test the CVP with real life case studies, which would include models that are more complex.
- Use other BIM authoring software (rather than Revit software) to integrate and run the developed prototype and generate the extended IFC files.
- Increase the building information in the CVP that deals with the changes of different elements and features. The current CVP is tracking the changes in the main building elements (beams, and columns). Further research is needed on the shared building elements (e.g. slabs, walls, doors, windows, etc.…) and the unshared building elements (e.g. reinforced bars, HVAC ducts, plumbing pipe, etc.…).
- Separate the current information (original part) and the versioning information (extended part) in the IFC into two files so that their

information is linked together in a way that the versioning information in one file is updated based on the new information in the other file.

- Study the interaction and communication among designers to deal with the design changes. This includes the response of each designer to each design change and building a communication among designers to accept or reject the changes. This can be done by suggesting new entities to deal with the communication issues or adopting BIM Collaboration Format (BCF) and build it into CVP.
- The proposed method do not only keep track over linear history, but also can model branches and manage merges. The CVP can be extended to support separating the same model version into branches (e.g. V_{1a}, V_{1b}) and to merge different sub- model versions.

8.6 Research Dissemination

The following papers have been published in which the author is named:

- JALY-ZADA, A., & TIZANI, W. 2013. **Building Information Modelling for Improving the Collaborative Design**. *15th Young Researchers' Conference. Institution of Structural Engineers*. London, UK.
- JALY-ZADA, A., & TIZANI, W. 2013. **Collaboration in Building Information Modelling**. *1st Kurdish Students Conference*. Nottingham, UK.
- JALY-ZADA, A., & TIZANI, W. 2013. **Multidisciplinary Collaboration in Building Information Modelling (BIM)**. *1st Kurdish Students Conference*. Nottingham, UK.
- JALY-ZADA, A., & TIZANI, W. 2013. **Collaboration between Architectural and Structural Models**. *11th International postgraduate research conference (IPGRC)*. University of Salford. Manchester, UK.
- JALY-ZADA, A., & TIZANI, W. 2013. **BIM for Improving Collaborative Design**. *East Midlands Universities/ Postgraduate Research Student Conference*. Derby, UK.

- JALY-ZADA, A., & TIZANI, W. 2014. **Multidisciplinary Collaborative Design Using Building Information Modelling – Versioning**. *16th Young Researchers' Conference. Institution of Structural Engineers*. London, UK.
- JALY-ZADA, A., TIZANI, W. & Oti, A.H. 2014. **Building Information Modelling (BIM) – Versioning for collaborative design**. *International Conference on Computing in Civil and Building Engineering. (ICCCBE2014)*. Florida, USA.
- Oti, A.H., TIZANI, W. &. JALY-ZADA, A. 2014. **A BIM extension for sustainability appraisal of conceptual structural design**. *International Conference on Computing in Civil and Building Engineering (ICCCBE2014)*. Florida, USA.
- JALY-ZADA, A., & TIZANI, W. 2014. **BIM to manage design changes.** *2nd Kurdish Students Conference*. Nottingham, UK.
- JALY-ZADA, A., KOCH, C., & TIZANI, W. 2015. **IFC Extension for Design Change Management**. *32rd international CIB W78 conference*. Eindhoven, Netherlands.

The author is currently involved in writing a set of journal publications. Following are proposal titles of the journal papers with brief bulleted descriptions about each of them:

- **IFC Extension for managing the design changes.**
  - Extend the IFC EXPRESS schema to deal with the versioning concept.
  - Implement the extended IFC in IFC-STEP file.
- **IFC based versioning for collaborative design.**
  - Use local and global versions.
  - Participate different disciplinary teams and use multiple BIM models.
  - Develop a prototype to deal with the extended IFC standard.
- **Design Change Management Based on Versioning the IFC Models.**
  - Develop change evolution graph.
  - Retrieve the whole previous information.
  - Use different version levels (model, element, and feature).
- **Multi-disciplinary collaboration in the cloud using a unique IFC file.**
  - Multiple teams work in the network.
  - Share a single and a unique IFC file.
  - Use local and global workspaces.

8.7 Summary

This research is proposed on the backdrop that utilizing the building information model to build an integrated engineering information environment to manage the design changes can contribute to support the collaborative design process. The practicality and efficiency of the suggested extension to the current IFC standard to cope with the changes in multiple BIM models could be positively assessed. It provides consistency and robustness to the collaborative design process through managing all new changes, all previous versions, and all version levels (model, element, and feature). Moreover, it supports versioning within the file exchange format (IFC) and visualizing all changes in the information model. A new way of working collaboratively among different disciplinary teams have been possible with the developed Collaboration Versioning Prototype (CVP).

References

(BIS), D. O. B. I. A. S. 2011. A report for the Government Construction Client Group , Building Information Modelling (BIM) Working Party Strategy Paper.

ADAMSON, D. A. P., T. 2006. *Change in the Construction Industry*

An Account of the UK Construction Industry Reform Movement 1993-2003, UK, Abingdon.

AKCAMETE, A., AKINCI, B. & GARRETT, J. H. Motivation for computational support for updating building information models (BIMs). Proceedings of the 2009 ASCE International Workshop on Computing in Civil Engineering, 2009. 523-532.

AKCAMETE, A., AKINCI, B. & GARRETT JR, J. H. Towards a formal approach for updating building information models. Proceedings of the 5th International Conference on Innovation in Architecture, Engineering and Construction, 2008.

AKINCI, B. & BOUKAMP, F. 2003. Representation and integration of as-built information to IFC based product and process models for automated assessment of as-built conditions.

AL HATTAB, M. & HAMZEH, F. Information flow comparison between traditional and BIM-based projects in the design phase. Annual Conference of the International Group for Lean Construction, Fortaleza, Brazil 21, 2013. 761-770.

ALARCÓN, L. F. & MARDONES, D. A. Improving the design-construction interface. Proceedings of the 6th Annual Meeting of the International Group for Lean Construction, 1998.

ALTMANNINGER, K., SEIDL, M. & WIMMER, M. 2009. A survey on model versioning approaches. *International Journal of Web Information Systems,* 5, 271-304.

AMARNATH, C., SAWHNEY, A. & MAHESWARI, J. U. Cloud computing to enhance collaboration, coordination and communication in the construction industry. Information and Communication Technologies (WICT), 2011 World Congress on, 2011. IEEE, 1235-1240.

AMOR, R. & GE, C. Mapping IFC versions. Proc of the EC-PPM Conference on eWork and eBusiness in AEC, Portoroz, Slovenia, 2002. 9-11.

AMOR, R., JIANG, Y. & CHEN, X. BIM in 2007–are we there yet. Proceedings of CIB W78 conference on Bringing ITC knowledge to work, Maribor, Slovenia, 2007. 26-29.

ANUMBA, C., UGWU, O., NEWNHAM, L. & THORPE, A. 2002. Collaborative design of structures using intelligent agents. *Automation in Construction,* 11, 89-103.

APACHE-PORTABLE-RUNTIME 2009. Versioning Numbering Concepts - The Apache Portable Runtime Project.

ARAYICI, Y. 2015. introduction to BIM.

ARNOLD, F. & PODEHL, G. 1999. Best of both worlds–a mapping from EXPRESS-G to UML. *The Unified Modeling Language.«UML»'98: Beyond the Notation.* Springer.

ASHCRAFT, H. W. 2009. BUILDING INFORMATION MODELING: A FRAMEWORK FOR COLLABORATION. *International Conference of Construction Law.* London, UK

ASLANI, P., GRIFFIS, F. & CHIARELLI, L. Building information model: The role and need of the constructors. Construction Research Congress 2009@ sBuilding a Sustainable Future, 2009. ASCE, 467-476.

AUTODESK. 2012. Unit 2 BIM Design Process. *http://bimcurriculum.autodesk.com/unit/unit-2-%E2%80%93-bim-design-process* [Online].

AUTODESK 2013. Revit 2013 API Developer's Guide.

AZHAR, S., CARLTON, W. A., OLSEN, D. & AHMAD, I. 2011. Building information modeling for sustainable design and LEED® rating analysis. *Automation in Construction,* 20, 217-224.

AZHAR, S., NADEEM, A., MOK, J. Y. & LEUNG, B. H. Building Information Modeling (BIM): A new paradigm for visual interactive modeling and simulation for construction projects. Proc., First International Conference on Construction in Developing Countries, 2008. 435-446.

BAKIS, N., AOUAD, G. & KAGIOGLOU, M. 2007. Towards distributed product data sharing environments — Progress so far and future challenges. *Automation in Construction,* 16, 586-595.

BARLISH, K. & SULLIVAN, K. 2012. How to measure the benefits of BIM—A case study approach. *Automation in Construction,* 24, 149-159.

BCIS, B. C. I. S. 2011. RICS 2011 Building Information Modelling Survey Report. *In:* SURVEYORS, R. I. O. C. (ed.).

BEETZ, J., DE LAAT, R., VAN BERLO, L. & VAN DEN HELM, P. Towards an open building information model server. Proc. of the 10th International Conference on Design & Decision Support Systems in Architecture and Urban Planning, The Netherlands, 2010.

BEUCKE, K. E. 2006. Versioned objects as a basis for engineering cooperation. *Intelligent Computing in Engineering and Architecture.* Springer.

BEWS, M. 2012. *Demystifying BIM* [Online]. USA: http://www.fenwickelliott.co.uk/files/insight_issue_7.pdf.

BIJL, A. & SHAWCROSS, G. 1975. Housing site layout system. *Computer-Aided Design,* 7, 2-10.

BIMSERVER. 2015. *http://bimserver.org/about/base-features/* [Online].

BIS 2011. A report for the Government Construction Client Group , Building Information Modelling (BIM) Working Party Strategy Paper. *Department of Business*.

BOEYKENS, S. 2012. Bridging building information modeling and parametric design. *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2012*, 453.

BOSAK, J. & BRAY, T. 1999. XML and the second-generation Web. *Scientific American,* 280, 89-93.

BOUCHLAGHEM, D. 2012. *Collaborative Working in Construction* SPON press.

BRUCE A. BURT, P. E. December 2009. BIM Interoperability , The Promise and the Reality. *STRUCTURE magazine.* USA: Council of American Engineers Associations.

BS-1192 2007. Collaborative production of architectural, engineering and construction information- Code of practice. *BS-1192.* UK: BSI 2007.

BSI 2013. Specifi cation for information management for the capital/delivery phase of construction projects using building information modelling. *PAS 1192-2:2013.* UK: The British Standards Institution 2013.

BUILDING-SMART 2015. http://www.buildingsmart-tech.org/mvd/review/IFC4Add1/DTVRV/candidate/html/.

BUILDINGSMART. 2015. *http://www.buildingsmart-tech.org/specifications/bcf-releases/bcf-intro* [Online].

BUILDINGSMART. 2016. *All Applications by Category* [Online]. 2016].

BUYYA, R., BROBERG, J. & GOSCINSKI, A. M. 2010. *Cloud computing: principles and paradigms*, John Wiley & Sons.

CABINET-OFFICE 2011. government construction strategy.

CAHILL, B., MENZEL, K. & FLYNN, D. 2012. BIM as a centre piece for optimized building operation. Taylor & Francis Group, London.

CALLISTER, W. D. & RETHWISCH, D. G. 2007. *Materials science and engineering: an introduction*, Wiley New York.

CDC 2011. Developing an Effective Evaluation Plan.

CDP. 2014. *CPD 14 2014: BIM - collaboration and the common data environment* [Online].

CELLARY, W. & JOMIER, G. Consistency of Versions in Object-Oriented Databases. VLDB, 1990. 432-441.

CEMESOVA, A. 2013. *Enhancing BIM-based data transfer to support the design of low energy buildings.* Cardiff University.

CEROVSEK, T. 2011. A review and outlook for a 'Building Information Model'(BIM): A multi-standpoint framework for technological development. *Advanced Engineering Informatics,* 25, 224-244.

CHAE, L. S. 2015. *Questionnaire for Owners to Evaluate Contractors' Proficiency in BIM.*

CHAO, T.-J. & KIM, Y. 2015. Interoperability format translation and transformation between IFC architectural design file and simulation file formats. Google Patents.

CHEN, H.-M. & HOU, C.-C. 2011. MULTI-DISCIPLINARY DESIGN COLLABORATION FOR DEVELOPING A BIM MODEL USING A HYBRID CLIENT-SERVER AND P2P NETWORK MODEL. *Proceedings of the 28th ISARC*, 1061-1066.

CHEN, H.-M. & HOU, C.-C. 2014. Asynchronous online collaboration in BIM generation using hybrid client-server and P2P network. *Automation in Construction,* 45, 72-85.

CHEN, P.-H., CUI, L., WAN, C., YANG, Q., TING, S. K. & TIONG, R. L. K. 2005a. Implementation of IFC-based web server for collaborative building design between architects and structural engineers. *Automation in Construction,* 14, 115-128.

CHEN, X., FUH, J., WONG, Y., LU, Y., LI, W. & QIU, Z. 2005b. An adaptable model for distributed collaborative design. *Computer-Aided Design and Applications,* 2, 47-55.

CHUANG, T.-H., LEE, B.-C. & WU, I.-C. Applying cloud computing technology to BIM visualization and manipulation. 28th International Symposium on Automation and Robotics in Construction, 2011. 144-149.

CINELIS, G. 2015. Review of the book by Richard Garber "BIM Design. Realising the Creative Potential of Building Information Modelling". *Journal of Sustainable Architecture and Civil Engineering,* 10, 99-101.

CLARKSON, J. & ECKERT, C. 2004. *Design process improvement: a review of current practice*, Springer.

CONRADI, R. & WESTFECHTEL, B. 1998. Version models for software configuration management. *ACM Computing Surveys (CSUR),* 30, 232-282.

CORONEL, C. & MORRIS, S. 2016. *Database Systems: Design, Implementation, & Management*, Cengage Learning.

CRNKOVIC, I., ASKLUND, U. & DAHLQVIST, A. P. 2003. *Implementing and integrating product data management and software configuration management*, Artech House.

DAWOOD, N. & IQBAL, N. 2010. Building Information Modelling: Scope for Innovation in the AEC Industry.

DAWOOD, S., CROSBIE, T., DAWOOD, N. & LORD, R. 2013. Designing low carbon buildings: A framework to reduce energy consumption and embed the use of renewables. *Sustainable Cities and Society,* 8, 63-71.

DEITEL, P. J., DEITEL, H. M. & DEITEL, A. 2013. *C# 2012 for Programmers*, Pearson Education.

DEMKIN, J. A. 2001. *The architect's handbook of professional practice*, John Wiley & Sons.

DENG, M. & DI, L. 2013. *Building open environments to meet big data challenges in Earth sciences*, Boca Raton, FL, USA: CRC Press.

DENG, X. Y. & CHANG, T.-Y. P. Creating structural model from IFC-based architectural model. Joint International Conference on Computing and Decision Making in Civil and Building Engineering, 2006. 3687-3695.

DEUTSCH, R. 2011. *BIM and Integrated Design: Strategies for Architectural Practice*, Wiley.

DING, L. & XU, X. 2014. Application of Cloud Storage on BIM Life-cycle Management. *International Journal of Advanced Robotics Systems, 11,* 129.

DONG, B., LAM, K., HUANG, Y. & DOBBS, G. A comparative study of the IFC and gbXML informational infrastructures for data exchange in computational design support environments. Tenth International IBPSA Conference, 2007. 1530-1537.

DYERGRIMES 2010. Stages of the Design & Building Process. *In:* ARCHITECTS, A. I. O. (ed.). AUSTRALIAN.

EASTERBROOK, S. & CALLAHAN, J. 1998. Formal methods for verification and validation of partial specifications: A case study. *Journal of Systems and Software,* 40, 199-210.

EASTMAN, C. & AUGENBROE, G. 1998. Product modeling strategies for today and the future. *CIB REPORT*, 191-208.

EASTMAN, C., EASTMAN, C. M., TEICHOLZ, P. & SACKS, R. 2011a. *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*, John Wiley & Sons.

EASTMAN, C. M., TEICHOLZ, P., SACKS, R. & LISTON, K. 2011b. *BIM handbook : a guide to building information modeling for owners, managers, designers,*

engineers and contractors - 2nd ed., Wiley Online Library.

ERIKSSON, H.-E. & PENKER, M. 2000. Business modeling with UML. *Business Patterns at Work, John Wiley & Sons, New York, USA*.

FAHDAH, I. 2008. *Distributed IT for integration and communication of engineering information for collaborative building design.* University of Nottingham.

FAHDAH, I. & TIZANI, W. Communication and concurrency approach for a real-time collaborative building design environment. 2008.

FILIPPI, S. & CRISTOFOLINI, I. 2009. *The design guidelines collaborative framework: a design for Multi-X method for product development*, Springer Science & Business Media.

FIRMENICH, B. A novel modelling approach for the exchange of CAD information in civil engineering. eWork and eBusiness in Architecture, Engineering and Construction: Proceedings of the 5th European Conference on Product and Process Modelling in the Building and Construction Industry-ECPPM 2004, 8-10 September 2004, Istanbul, Turkey, 2004. Taylor & Francis, 123.

FIRMENICH, B. 2005. Consistency of a Shared Versioned Model for Distributed Cooperation. *Computer-Aided Civil and Infrastructure Engineering,* 20, 424-430.

FIRMENICH, B., KOCH, C., RICHTER, T. & BEER, D. G. 2005. Versioning structured object sets using text based Version Control Systems. *Proceedings of the 22nd CIB-W78. Institute of Construction Informatics, Dresden*.

FLURL, M., MUNDANI, R., BORRMANN, A. & RANK, E. A collaborative multi-scale planning platform: Concept and implementation approach. Proc. of the International Conference on Computing in Civil and Building Engineering, Moscow, Russia, 2012.

GALLAHER, M. P., O'CONNO, A. C., DETTBARN, J. L. & GILDAY, L. T. J. 2004. Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry. National Institute of Standards and Technology, Gaithersburg, MD

GANAH, A. & JOHN, G. A. 2014. Achieving Level 2 BIM by 2016 in the UK. *Computing in Civil and Building Engineering,* 2, 1.

GANAH, A. & JOHN, G. A. 2015. An Overview of the Feasibility of Achieving Level 2 Building Information Modeling by 2016 in the UK.

GAO, G., LIU, Y.-S., WANG, M., GU, M. & YONG, J.-H. 2015. A query expansion method for retrieving online BIM resources based on Industry Foundation Classes. *Automation in Construction,* 56, 14-25.

GAREIS, R. 2010. Changes of organizations by projects. *International Journal of Project Management,* 28, 314-327.

GARG, S. K., VERSTEEG, S. & BUYYA, R. 2013. A framework for ranking of cloud computing services. *Future Generation Computer Systems,* 29, 1012-1023.

GEDIGA, G., HAMBORG, K.-C. & DÜNTSCH, I. 2002. Evaluation of software systems. *Encyclopedia of computer science and technology,* 45, 127-53.

GEIGER, A., BENNER, J. & HAEFELE, K. H. 2015. Generalization of 3D IFC Building Models. *3D Geoinformation Science.* Springer.

GERYVILLE, H., OUZROUT, Y., BOURAS, A. & SAPIDIS, N. 2007. A collaborative framework to exchange and share product information within a supply chain context. *arXiv preprint arXiv:0706.0507*.

GIELINGH, W. 2008. An assessment of the current state of product data technologies. *Computer-Aided Design,* 40, 750-759.

GILLHAM, B. 2000. *Case study research methods*, Bloomsbury Publishing.

GIRARD, P. & ROBIN, V. 2006. Analysis of collaboration for project design management. *Computers in Industry,* 57, 817-826.

GÖKÇE, K. U., GÖKÇE, H. U. & KATRANUSCHKOV, P. 2012. IFC-based product catalog formalization for software interoperability in the construction management domain. *Journal of computing in civil engineering,* 27, 36-50.

GÖKÇE, K. U., GÖKÇE, H. U. & KATRANUSCHKOV, P. 2013. IFC-Based Product Catalog Formalization for Software Interoperability in the Construction Management Domain. *Journal of computing in civil engineering,* 27, 36-50.

GONNET, S., HENNING, G. & LEONE, H. 2007. A model for capturing and representing the engineering design process. *Expert Systems with Applications,* 33, 881-902.

GRAPHISOFT. 2015. *http://helpcenter.graphisoft.com/guides/* [Online].

GRILO, A. & JARDIM-GONCALVES, R. 2010. Value proposition on interoperability of BIM and collaborative working environments. *Automation in Construction,* 19, 522-530.

GRILO, A. & JARDIM-GONCALVES, R. 2011. Challenging electronic procurement in the AEC sector: A BIM-based integrated perspective. *Automation in Construction,* 20, 107-114.

GU, N., KIM, M. J. & MAHER, M. L. 2011. Technological advancements in synchronous collaboration: The effect of 3D virtual worlds and tangible user interfaces on architectural design. *Automation in Construction,* 20, 270-278.

GU, N. & LONDON, K. 2010. Understanding and facilitating BIM adoption in the AEC industry. *Automation in Construction,* 19, 988-999.

GUPTA, A., CEMESOVA, A., HOPFE, C. J., REZGUI, Y. & SWEET, T. 2014. A conceptual framework to support solar PV simulation using an open-BIM data exchange standard. *Automation in Construction,* 37, 166-181.

GYSSENS, M., PAREDAENS, J., VAN DEN BUSSCHE, J. & VAN GUCHT, D. 1994. A graph-oriented object database model. *Knowledge and Data Engineering, IEEE Transactions on,* 6, 572-586.

HAO, Q., SHEN, W., NEELAMKAVIL, J. & THOMAS, R. 2008. Change management in construction projects. *NRC Institute for Research in Construction, NRCC-50325*.

HARDIN, B. & MCCOOL, D. 2015. *BIM and construction management: proven tools, methods, and workflows*, John Wiley & Sons.

HASS, G. 2003. *Configuration management principles and practice*, Addison-Wesley Longman Publishing Co., Inc.

HELM, P., BOHMS, H. & BERLO, L. IFC-based clash detection for the open-source BIMserver. ICCBE Conferentie, 7, 2010.

HM-GOVERNMENT 2013. Industrial Strategy: government and industry in partnership. (Construction 2025)

HOLZER, D. 2015. *The BIM Manager's Handbook, Part 2: Change Management*, John Wiley & Sons.

HOLZINGER, A. 2005. Usability engineering methods for software developers. *Communications of the ACM,* 48, 71-74.

HOOSHYAR, D., AHMAD, R. B., NASIR, M. H. N. M. & MUN, W. C. Flowchart-based approach to aid novice programmers: A novel framework. Computer and Information Sciences (ICCOINS), 2014 International Conference on, 2014. IEEE, 1-5.

HOPFE, C. J., STRUCK, C., HARPUTLUGIL, G. U., HENSEN, J. & DE WILDE, P. Exploration of the use of building performance simulation for conceptual design. IBPSA-NVL Conference, 2005. 1-16.

HOWARD, R. & BJÖRK, B.-C. 2008. Building information modelling–Experts' views on standardisation and industry deployment. *Advanced Engineering Informatics,* 22, 271-280.

IBBS, C. W., WONG, C. K. & KWAK, Y. H. 2001. Project change management system. *Journal of Management in Engineering*.

ICAP. 2012. *What Is Evaluation?* [Online].

IFCWIKI 2015. http://www.ifcwiki.org/index.php/Free_Software.

IGNATOVA, E., KIRSCHKE, H., TAUSCHER, E. & SMARSLY, K. 2015. PARAMETRIC GEOMETRIC MODELING IN CONSTRUCTION PLANNING USING INDUSTRY FOUNDATION CLASSES. *Applications of Computer Science and Mathematics in Architecture and Civil Engineering*, 68.

ISAAC, S. & NAVON, R. 2009. Modeling building projects as a basis for change control. *Automation in Construction,* 18, 656-664.

ISAAC, S. & NAVON, R. 2013. A graph-based model for the identification of the impact of design changes. *Automation in Construction,* 31, 31-40.

ISIKDAG, U. 2012. Design patterns for BIM-based service-oriented architectures. *Automation in Construction,* 25, 59-71.

ISO 2013. Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries.

JACKO, J. A. 2012. *Human Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications*, CRC press.

JALY-ZADA, A., KOCH, C. & TIZANI, W. IFC EXTENSION FOR DESIGN CHANGE MANAGEMENT. 32rd CIB W78 conference 2015, 2015 Netherlands. CIB W78.

JALY-ZADA, A. & TIZANI, W. 2013. Collaboration between Architectural and Structural Models. *11th. International Postgraduate Research Conference (IPGRC13).* University of Salford / Manchester.

JALY-ZADA, A. & TIZANI, W. 2014. Multidisciplinary Collaborative Design Using Building Information Modelling – Versioning. *In:* ENGINEERS, I. O. S. (ed.) *Young Researchers' Conference.* London, UK.

JALY-ZADA, A., TIZANI, W. & OTI, A. Building Information Modelling (BIM)—Versioning for Collaborative Design. Computing in Civil and Building Engineering (2014), 2014 USA. ASCE, 512-519.

JENA, A. K., SWAIN, S. K. & MOHAPATRA, D. P. 2015. Model Based Test Case Generation from UML Sequence and Interaction Overview Diagrams. *Computational Intelligence in Data Mining-Volume 2.* Springer.

JI, Y., BEETZ, J., BONSMA, P., BISBET, N., KATZ, C. & BORRMANN, A. Integration of parametric geometry into IFC-Bridge. Proc. of the 23th Forum Bauinformatik, Cork, Ireland, 2011.

JIAO, Y., WANG, Y., ZHANG, S., LI, Y., YANG, B. & YUAN, L. 2013. A cloud approach to unified lifecycle data management in architecture, engineering, construction and facilities management: Integrating BIMs and SNS. *Advanced Engineering Informatics,* 27, 173-188.

JOHANSEN, R. 1988. *GroupWare: Computer Support for Business Teams,* New York, NY, USA

KARAN, E. P. & IRIZARRY, J. 2015. Extending BIM interoperability to preconstruction operations using geospatial analyses and semantic web services. *Automation in Construction,* 53, 1-12.

KARSTILA, K., SERÉN, K. & OY, E. IFC Release 2.0 Certification testing–IFC Schema Refinement. IAI Forum Finland/BLIS, 2001. 35.

KASSEM, M., IQBAL, N., KELLY, G., LOCKLEY, S. & DAWOOD, N. 2014. Building information modelling: protocols for collaborative design processes. *Journal of Information Technology in Construction (ITcon),* 19, 126-149.

KENSEK, K. M. 2014. *Building Information Modeling*, Routledge.

KIM, S., CHOE, Y., JANG, M. & SEOL, W. 2011. Design process visualization system integrating BIM data and performanceoriented design information. *Proceedings of the 28th ISARC, Seoul, Korea*.

KITAGAWA, H., GOTOH, M., MISAKI, S. & AZUMA, M. Form document management system SPECDOQ its architecture and implementation. ACM SIGOA Newsletter, 1984. ACM, 132-142.

KLÉOS , J., RITA , A. & JOSÉ , C. IFC and the Representation of Reinforced Concrete Structures (Part 2)

An Approach to Load Combinations on Beams ANAIS DO 54° CONGRESSO BRASILEIRO DO CONCRETO - CBC2012 — 54CBC, 2012 Brasil. IBRACON.

KOCH, C. & FIRMENICH, B. A novel diff and merge approach on the basis of operative models. Proc. of the 11th Int. Conf. on Civil and Building Engineering (ICCCBE-XI), 2006.

KOCH, C. & FIRMENICH, B. 2011. An approach to distributed building modeling on the basis of versions and changes. *Advanced Engineering Informatics,* 25, 297-310.

KOEGEL, M., HERRMANNSDOERFER, M., LI, Y., HELMING, J. & DAVID, J. Comparing state-and operation-based change tracking on models. Enterprise Distributed

Object Computing Conference (EDOC), 2010 14th IEEE International, 2010. IEEE, 163-172.

KOSKELA, L. 1992. *Application of the new production philosophy to construction*, Stanford university Stanford, CA.

KYMMELL, W. 2008. *Building Information Modeling: Planning and Managing Construction Projects with 4D CAD and Simulations (McGraw-Hill Construction Series): Planning and Managing Construction Projects with 4D CAD and Simulations*, McGraw Hill Professional.

LAAKSO, M. & KIVINIEMI, A. 2012. The IFC standard: A review of History, development, and standardization, Information Technology. *ITcon,* 17, 134-161.

LANGDON, D. 2012. Getting the most out of BIM , A guide for clients. *In:* LANGDON, D. (ed.) *AECOM.*

LEA, G., GANAH, A., GOULDING, J. S. & AINSWORTH, N. 2015. Identification And Analysis Of UK And US BIM Standards To Aid Collaboration.

LEE, J., JEONG, Y., OH, M. & HONG, S. W. 2014. A Filter-Mediated Communication Model for Design Collaboration in Building Construction. *The Scientific World Journal,* 2014.

LEE, J., SMITH, J. & KANG, J. 2011. THE ROLE OF IFC FOR SUSTAINABLE BIM DATA MANAGEMENT.

LEE, S.-H. & KIM, B.-G. 2011. IFC extension for road structures and digital modeling. *Procedia Engineering,* 14, 1037-1042.

LEE, S.-I., BAE, J.-S. & CHO, Y. S. 2012. Efficiency analysis of Set-based Design with structural building information modeling (S-BIM) on high-rise building structures. *Automation in Construction,* 23, 20-32.

LI, W., LU, W., FUH, J. Y. H. & WONG, Y. 2005. Collaborative computer-aided design—research and development status. *Computer-Aided Design,* 37, 931-940.

LI, W. & QIU, Z. 2007. State-of-the-art technologies and methodologies for collaborative product development systems. *International Journal of Production Research,* 44, 2525-2559.

LIEBICH, T. 2009. IFC 2x Edition 3 Model Implementation Guide *In:* 2.0, V. (ed.). buildingSMART International Modeling Support Group

LIEBICH, T., WEISE, M., BÖHMS, M., BONSMA, P., FIES, B., BORDEAU, M., DUNWELL, S., PRATT, J. & ČUŠ, N. 2008. OPEN STANDARDS FOR INTEROPERABILITY BETWEEN APPLICATIONS IN EARLY DESIGN.

LIEBICH, T. & WIX, J. Highlights of the development process of industry foundation classes. Proceedings of the 1999 CIB W78 Conference, 1999. 18.

LINHARD, K. & STEINMANN, R. 2014. BIM-collaboration processes–from fuzziness to practical implementation. *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014*, 27.

LIPMAN, R. & LUBELL, J. 2015. Conformance checking of PMI representation in CAD model STEP data exchange files. *Computer-Aided Design,* 66, 14-23.

LITTLER, D., LEVERICK, F. & BRUCE, M. 1995. Factors affecting the process of collaborative product development: a study of UK manufacturers of information and communications technology products. *Journal of Product Innovation Management,* 12, 16-32.

LIU, D. T. & XU, X. W. 2001. A review of web-based product data management systems. *Computers in Industry,* 44, 251-262.

LOCK, M. D. & FLOURIS, T. G. 2012. *Managing Aviation Projects from Concept to Completion*, Ashgate Publishing, Ltd.

LOVE, P. E., EDWARDS, D. J., HAN, S. & GOH, Y. M. 2011. Design error reduction: toward the effective utilization of building information modeling. *Research in Engineering Design,* 22, 173-187.

MA, L., SACKS, R. & ZEIBAK-SHINI, R. 2015. Information modeling of earthquake-damaged reinforced concrete structures. *Advanced Engineering Informatics*.

MA, Z., WEI, Z. & ZHANG, X. 2013. Semi-automatic and specification-compliant cost estimation for tendering of building projects based on IFC data of design model. *Automation in Construction,* 30, 126-135.

MACDONALD, J. A. G., MARKO 2013. Multidisciplinary AEC Education Utilising BIM / PLIM Tools and Processes. *In:* KTH, S. O. A. A. T. B. E. A. (ed.) *PLM International Conference (PLM IC), PLM 13.* Nantes, France, 6th-10th July 2013.

MAHDAVI, A., MARTENS, B. & SCHERER, R. 2014. *eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014*, CRC Press.

MARADZA, E., WHYTE, J. & LARSEN, G. D. Interactive learning in UK construction practice: examining the role of BIM process standards. and Aboagye-Nimo, E (Ed.), Proceedings 30th Annual ARCOM Conference, 2014. 1-3.

MASONS, P. 2012. *Building Information Modelling* [Online].

MCELROY, S. & W.TIZANI, D. 2008. Process Mapping to encourage AE Collaboration during Design.

MCGRAW-HILL 2014. SmartMarket Report. Autodesk.

MCGRAW-HILL 2015. Business value of BIM|ANZ - Autodesk.

MCLAFFERTY, S. L. 2003. Conducting questionnaire surveys. *Key methods in geography*, 87-100.

MELL, P. & GRANCE, T. 2009. The NIST definition of cloud computing. *National Institute of Standards and Technology,* 53, 50.

MIKAEL LAAKSO, A. K. 2012. THE IFC STANDARD - A REVIEW OF HISTORY, DEVELOPMENT, AND STANDARDIZATION. *Journal of Information Technology in Construction,* 17, 134-161.

MILENTIJEVIC, I., CIRIC, V. & VOJINOVIC, O. 2008. Version control in project-based learning. *Computers & Education,* 50, 1331-1338.

MOKHTAR, A., BÉDARD, C. & FAZIO, P. 1998. Information model for managing design changes in a collaborative environment. *Journal of computing in civil engineering*.

MOTAMEDI, A., SOLTANI, M. M., SETAYESHGAR, S. & HAMMAD, A. 2016. Extending IFC to incorporate information of RFID tags attached to building elements. *Advanced Engineering Informatics,* 30, 39-53.

MOTAWA, I., ANUMBA, C., LEE, S. & PEÑA-MORA, F. 2007. An integrated system for change management in construction. *Automation in Construction,* 16, 368-377.

MOTAWA, I. A., ANUMBA, C. J. & EL-HAMALAWI, A. 2006. A fuzzy system for evaluating the risk of change in construction projects. *Advances in Engineering Software,* 37, 583-591.

MULLER, M. F., LOURES, E. R. & CANCIGLIERI, O. 2015. Interoperability Assessment for Building Information Modelling.

NAMINI, S. B., SHOURANGIZ, E. & HASSANABADI, M. S. 2011. Flexibility of BIM towards Design Change.

NAWARI, N. O. 2015. Synthesizing Aspects and Constraints of Structural Design Using BIM and a Proposed Framework in Education. *Building Information Modeling*, 1.

NBS 2013. NBS National BIM Report 2013. *In:* THIRD (ed.).

NBS. 2014. *BIM levels explained* [Online].

NBS 2015. NBS National BIM Report 2015. *In:* FIFTH (ed.).

NGO, M. H. 2012. uk construction industry's responses to government construction strategy BIM deadline and applications to civil engineering education. *In:* LONDON, I. C. (ed.) *1st. civil and environmental engineering student conference.* UK: Imperial Collage London.

NIBS 2007. National Building Information Modeling Standard. *Version 1: Part 1.*

NIZAM, R. & ZHANG, C. Current state of information exchange between the two most popular BIM software: Revit and Tekla. Sustainable Buildings and Structures: Proceedings of the 1st International Conference on Sustainable Buildings and Structures (Suzhou, PR China, 29 October-1 November 2015), 2015. CRC Press, 175.

NOUR, M. 2009a. A Graphical User Interface for handling IFC Partial Model Exchange.

NOUR, M. 2009b. Performance of different (BIM/IFC) exchange formats within private collaborative workspace for collaborative work. ITcon.

NOUR, M. 2012. BIM support for design change management and workflow approval *In:* ISCCBE (ed.) *14th International Conference on Computing in civil and Building Engineering.* Moscow , Russia

NOUR, M. & BEUCKE, K. 2008. An open platform for processing IFC model versions. *Tsinghua Science & Technology,* 13, 126-131.

NOUR, M. & BEUCKE, K. Object versioning as a basis for design change management within a BIM context. Proceedings of the 13th international conference on computing in civil and building engineering (ICCCBE-XIII), Nottingham, UK, 2010.

NOUR, M., FIRMENICH, B., RICHTER, T. & KOCH, C. A versioned IFC database for multidisciplinary synchronous cooperation. Proceedings of the Joint International Conference on Computing and Decision Making in Civil and Building Engineering, Montreal, Canada, 2006. 636-645.

OLOFSSON, T., LEE, G. & EASTMAN, C. 2008. Editorial-Case studies of BIM in use. *IT in construction-Special Issue Case studies of BIM use,* 13, 244-245.

ORVIZ, A. 2010. *Effective Collaboration in Multi-disciplinary Teams*.

OTI, A., TIZANI, W. & JALY-ZADA, A. A BIM Extension For Sustainability Appraisal Of Conceptual Structural Design Of Steel-Framed Buildings. Computing in Civil and Building Engineering (2014), 2014. ASCE, 219-226.

OTI, A. H. & TIZANI, W. 2014. BIM extension for the sustainability appraisal of conceptual steel design. *Advanced Engineering Informatics*.

OTI, A. H. & TIZANI, W. 2015. BIM extension for the sustainability appraisal of conceptual steel design. *Advanced Engineering Informatics,* 29, 28-46.

OXFORD.DICTIONARIES. 2012. *http://oxforddictionaries.com/definition/english/interoperable?q=interoperability#interoperable__2* [Online].

PAHL, G., BEITZ, W., SCHULZ, H. J. & JARECKI, U. 2007. *Engineering design: a systematic approach*, Springer.

PAYNE, T. 2013. *What is a Common Data Environment?* [Online].

PELTONEN, H., PITKÄNEN, O. & SULONEN, R. 1996. Process-based view of product data management. *Computers in Industry,* 31, 195-203.

PETERSON, F., HARTMANN, T., FRUCHTER, R. & FISCHER, M. 2011. Teaching construction project management with BIM support: Experience and lessons learned. *Automation in Construction,* 20, 115-125.

PILATO, C. M., COLLINS-SUSSMAN, B. & FITZPATRICK, B. W. 2008. *Version control with subversion*, " O'Reilly Media, Inc.".

PILEHCHIAN, B., STAUB-FRENCH, S. & NEPAL, M. P. 2015. A conceptual approach to track design changes within a multi-disciplinary building information modeling environment. *Canadian Journal of Civil Engineering,* 42, 139-152.

PLUME, J. & MITCHELL, J. 2007. Collaborative design using a shared IFC building model—Learning from experience. *Automation in Construction,* 16, 28-36.

PRATT, M. J. 2001. Introduction to ISO 10303—the STEP standard for product data exchange. *Journal of Computing and Information Science in Engineering,* 1, 102-103.

QUICK, D. & CHOO, K.-K. R. 2013. Dropbox analysis: Data remnants on user machines. *Digital Investigation,* 10, 3-18.

REDMOND, A., HORE, A., ALSHAWI, M. & WEST, R. 2012. Exploring how information exchanges can be enhanced through Cloud BIM. *Automation in Construction,* 24, 175-183.

REVIT. 2015. *http://www.autodesk.co.uk/products/revit-family/overview* [Online].

REZGUI, Y., BEACH, T. & RANA, O. 2013. A GOVERNANCE APPROACH FOR BIM MANAGEMENT ACROSS LIFECYCLE AND SUPPLY CHAINS USING MIXED-MODES OF INFORMATION DELIVERY. *Journal of Civil Engineering and Management,* 19, 239-258.

REZGUI, Y. & DIRECTOR, B. 2012. CLOUDBIM: MANAGEMENT OF BIM DATA IN A CLOUD COMPUTING ENVIRONMENT.

RICCI, A. 1973. A constructive geometry for computer graphics. *The Computer Journal,* 16, 157-160.

RICHTER, T. & BEUCKE, K. Diff and merge for net-distributed applications in civil engineering. Proceedings of the Eleventh International Conference (ICCCBE-XI), Montreal, 2006.

RICHTER, T. & BEUCKE, K. 2008. A concept for Utilizing versioned object models in engineering applications *12th International Conferences on Computing in Civil and Building Engineering & 2008 International Conference on Information Technology in Construction.* Beijing, China

RIDWAN, F., XU, X. & LIU, G. 2012. A framework for machining optimisation based on STEP-NC. *Journal of Intelligent Manufacturing,* 23, 423-441.

ROBERTSON, D. & ALLEN, T. J. 1993. CAD system use and engineering performance. *Engineering Management, IEEE Transactions on,* 40, 274-282.

RODDICK, J. F. 1995. A survey of schema versioning issues for database systems. *Information and Software Technology,* 37, 383-393.

ROLDÁN, M. L., GONNET, S. & LEONE, H. 2010. TracED: a tool for capturing and tracing engineering design processes. *Advances in Engineering Software,* 41, 1087-1109.

ROSCHELLE, J. & TEASLEY, S. D. The construction of shared knowledge in collaborative problem solving. Computer supported collaborative learning, 1995. Springer, 69-97.

ROSHANI, D. & TIZANI, W. Integrated IFC based collaborative building design using internet technology. CD, The Tenth International Conference on Civil, Structural and Environmental Engineering Computing, Rome, Italy, 2005.

ROSSI, P. H., LIPSEY, M. W. & FREEMAN, H. E. 2003. *Evaluation: A systematic approach*, Sage publications.

RUDDER, D. 2013. *Autodesk Revit 2013 customization with. NET how-to*, Packt Publ.

RUIKAR, D., SMITH, R., TIZANI, W. & NETHERCOT, D. 2001. Best Practice Approach in the Design of Multi Storey Steel Framed Structures.

RUIKAR, D., TIZANI, W. & SMITH, R. Design Process Improvement using a Single Model Environment. *In:* TOPPING, B. H. V., ed. PROCEEDINGS OF THE TENTH

INTERNATIONAL CONFERENCE ON CIVIL, STRUCTURAL AND ENVIRONMENTAL ENGINEERING COMPUTING, 2005 University of Nottingham / UK. University of Nottingham.

SAAD, M. & MAHER, M. L. 1996. Shared understanding in computer-supported collaborative design. *Computer-Aided Design,* 28, 183-192.

SAWHNEY, A. & MAHESWARI, J. U. 2013. Design coordination using cloud-based smart building element models. *International Journal of Computer Information Systems and Industrial Management Applications,* 5, 445-453.

SCAPENS, R. W. 2004. Doing case study research. *The real life guide to accounting research*, 257-279.

SCHENSUL, S. L., SCHENSUL, J. J. & LECOMPTE, M. D. 1999. *Essential ethnographic methods: Observations, interviews, and questionnaires*, Rowman Altamira.

SCHNEIDER, G. & WINTERS, J. P. 2001. *Applying use cases: a practical guide*, Pearson Education.

SCHULER, R. W. 2001. The Application of ISO 10303-11 (the EXPRESS Language) in Defining Data Models for Software Design and Implementation.

SHAFIQ, M. T., MATTHEWS, J. & LOCKLEY, S. 2013. A study of BIM collaboration requirements and available features in existing model collaboration systems. *Journal of Information Technology in Construction (ITcon),* 18, 148-161.

SHEN, W., HAO, Q. & LI, W. 2008. Computer supported collaborative design: Retrospective and perspective. *Computers in Industry,* 59, 855-862.

SHEN, W., HAO, Q., MAK, H., NEELAMKAVIL, J., XIE, H., DICKINSON, J., THOMAS, R., PARDASANI, A. & XUE, H. 2010. Systems integration and collaboration in architecture, engineering, construction, and facilities management: A review. *Advanced Engineering Informatics,* 24, 196-207.

SHOURANGIZ, E., MOHAMAD, M. I., HASSANABADI, M. S. & SAEED, S. Flexibility of BIM towards Design Change. 2nd International Conference on Construction and Project Management, IPEDR vol. 15 (2011)©(2011) IACSIT Press, Singapore, 2011.

SHYAMSUNDAR, N. & GADH, R. 2002. Collaborative virtual prototyping of product assemblies over the Internet. *Computer-Aided Design,* 34, 755-768.

ŠILINGAS, D. & BUTLERIS, R. 2015. Towards implementing a framework for modeling software requirements in MagicDraw UML. *Information Technology and Control,* 38.

SINCLAIR, D. 2013. RIBA plan of work 2013 overview. London: Royal Institute of British Architects.

SINGH, D., THAKUR, A. & CHAUDHARY, A. 2015. A Comparative Study between Waterfall and Incremental Software Development Life Cycle Model. *International Journal of Emerging Trends in Science and Technology,* 2.

SINGH, V., GU, N. & WANG, X. 2011. A theoretical framework of a BIM-based multi-disciplinary collaboration platform. *Automation in Construction,* 20, 134-144.

SMITH, D. 2010. Optimazing Performance and Profitability BIM. Building Smart Alliance: Int. Alliance for Interoperability.

SMITH, R. A. & TIZANI, W. 2006. The Role of VR in Multi Discipline Building Design.

SOLIBRI 2014. Getting Started with Solibri Model Checker™.

SOLIHIN, W. & EASTMAN, C. 2015. Classification of rules for automated BIM rule checking development. *Automation in Construction,* 53, 69-82.

SPINELLIS, D. 2005. Version control systems. *Software, IEEE,* 22, 108-109.

STAKE, R. E. 2013. *Multiple case study analysis*, Guilford Press.

STEEL, J., DROGEMULLER, R. & TOTH, B. 2012a. Model interoperability in building information modelling. *Softw. Syst. Model.,* 11, 99-109.

STEEL, J., DROGEMULLER, R. & TOTH, B. 2012b. Model interoperability in building information modelling. *Software and Systems Modeling,* 11, 99-109.

SUCCAR, B. 2009. Building information modelling framework: A research and delivery foundation for industry stakeholders. *Automation in Construction,* 18, 357-375.

SUN, J., LIU, Y.-S., GAO, G. & HAN, X.-G. 2015. IFCCompressor: A content-based compression algorithm for optimizing Industry Foundation Classes files. *Automation in Construction,* 50, 1-15.

SUN, M. & AOUAD, G. Integration Technologies to Support Organisational Changes in the Construction Industry. 7th ISPE International Conference on Concurrent Engineering, Lyon, France,, 2000. pp 596.

SUN, M. & MENG, X. 2009. Taxonomy for change causes and effects in construction projects. *International Journal of Project Management,* 27, 560-572.

TAENTZER, G., ERMEL, C., LANGER, P. & WIMMER, M. 2014. A fundamental approach to model versioning based on graph modifications: from theory to implementation. *Software & Systems Modeling,* 13, 239-272.

TAI, P.-K. & ZOU, H. 1996. Light expanding system for producing a linear or planar light beam from a point-like light source. Google Patents.

TAYLOR, C. 2009. Collaboration Platform. *In:* INNOVATION, C. C. (ed.) *Research Project No: 2007-003-EP.* AUSTRALIA.

TEKLA-COMPANY. 2015. *http://www.bigprojectme.com/news/autodesk-to-launch-specialised-bim-steel-software-in-2015/attachment/francis-crick-bim-show-live/* [Online].

THOMASSEN, M. 2011. BIM & Collaboration in the AEC Industry. *Construction Management, Master's (MSc) Thesis (Released 31 August 2011, p. 120, in: http://projekter. aau. dk/projekter/files/55376698/BIM_Collaboration_in_the_AEC_Industry_by_Mats_Thomassen. pdf*.

TIZANI, W. 2011. Collaborative Design in Virtual Environments. Springer-Verlag New York Inc.

TIZANI, W. & MAWDESLEY, M. J. 2011. Advances and challenges in computing in civil and building engineering. *Advanced Engineering Informatics,* 25, 569-572.

TIZANI, W., SMITH, R. & RUIKAR, D. Decision support through an integrated product and process model. Proceedings 3rd International Conference on Decision Making in Urban & Civil Engineering (DminUce), 2002 London, UK.

TOMASOWA, R. 2015. BIM Design Collaboration Report.

TUNSTALL, G. 2006. *Managing the building design process*, Routledge.

TURNER, J. R. 2014. *The handbook of project-based management*, McGraw-hill.

UNDERWOOD, J. 2009. *Handbook of Research on Building Information Modeling and Construction Informatics: Concepts and Technologies: Concepts and Technologies*, IGI Global.

US.PRO 2006. Initial Graphics Exchange Specification: IGES 5.3.

VAN BERLO, L. & KRIJNEN, T. 2014. Using the BIM Collaboration Format in a Server Based Workflow. *Procedia Environmental Sciences,* 22, 325-332.

VESPERMAN, J. 2006. *Essential CVS*, " O'Reilly Media, Inc.".

VOLK, R., STENGEL, J. & SCHULTMANN, F. 2014. Building Information Modeling (BIM) for existing buildings—Literature review and future needs. *Automation in Construction,* 38, 109-127.

WANG, H., AKINCI, B. & GARRETT JR, J. H. 2007. Formalism for detecting version differences in data models. *Journal of computing in civil engineering,* 21, 321-330.

WANG, H., AKINCI, B., GARRETT JR, J. H., NYBERG, E. & REED, K. A. 2009. Semi-automated model matching using version difference. *Advanced Engineering Informatics,* 23, 1-11.

WANG, H., JOHNSON, A. L. & BRACEWELL, R. H. 2012a. The retrieval of structured design rationale for the re-use of design knowledge with an integrated representation. *Advanced Engineering Informatics,* 26, 251-266.

WANG, H., SHEA, R., WANG, F. & LIU, J. On the impact of virtualization on dropbox-like cloud file storage/synchronization services. Proceedings of the 2012 IEEE 20th international workshop on quality of service, 2012b. IEEE Press, 11.

WANG, L., SHEN, W., XIE, H., NEELAMKAVIL, J. & PARDASANI, A. 2002. Collaborative conceptual design—state of the art and future trends. *Computer-Aided Design,* 34, 981-996.

WANG, X., YANG, H. & ZHANG, Q. 2014. Research of the IFC-based Transformation Methods of Geometry Information for Structural Elements. *Journal of Intelligent & Robotic Systems*, 1-9.

WANG, X. V. & XU, X. W. 2015. A collaborative product data exchange environment based on STEP. *International Journal of Computer Integrated Manufacturing,* 28, 75-86.

WEISE, M., KATRANUSCHKOV, P. & SCHERER, R. 2000. A proposed extension of the IFC project model for structural systems. *Proc. Product and Process Modelling in Building and Construction, Rotterdam*.

WEISE, M., LIEBICH, T. & WIX, J. 2009. Integrating use case definitions for IFC developments. *eWork and eBusiness in Architecture and Construction. London: Taylor & Francis Group*, 637-645.

WEYGANT, R. S. 2011. *BIM content development: standards, strategies, and best practices*, John Wiley & Sons.

WITTENBURG, T. & LITTLE, T. D. An adaptive document management system for shared multimedia data. Multimedia Computing and Systems, 1994., Proceedings of the International Conference on, 1994. IEEE, 245-254.

WIX, J., NISBET, N. & LIEBICH, T. 2008. Using constraints to validate and check building information models. *E-Work and EBuisness in Architecture, Engineering and Construction: ECPPM, London*, 467-475.

WONG, J., WANG, X., LI, H., CHAN, G. & LI, H. 2014. A review of cloud-based BIM technology in the construction sector. *The Journal of Information Technology in Construction,* 19, 281-291.

WU, D., ROSEN, D. W., WANG, L. & SCHAEFER, D. 2015. Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation. *Computer-Aided Design,* 59, 1-14.

WU, I. C. & HSIEH, S.-H. 2012. A framework for facilitating multi-dimensional information integration, management and visualization in engineering projects. *Automation in Construction,* 23, 71-86.

XIAOLONG, X., QIPING, S., HONGQIN, F., HENG, L. & SHICHAO, F. 2012. IT supported collaborative work in A/E/C projects: A ten-year review. *Autom Constr,* 21, 1-9.

XIAOLONG, X., YAOWU, W. & QIPING, S. The Application of ICT in Collaborative Working in Construction Projects: A Critical Review. 2007. IEEE, 2266-2271.

XUE, X., SHEN, Q. & REN, Z. 2010. Critical review of collaborative working in construction projects: Business environment and human behaviors. *Journal of Management in Engineering,* 26, 196.

YAO, K.-T., NECHES, R., KO, I.-Y., ELEISH, R. & ABHINKAR, S. Synchronous and asynchronous collaborative information space analysis tools. Parallel

Processing, 1999. Proceedings. 1999 International Workshops on, 1999. IEEE, 74-79.

YUE, T., BRIAND, L. C. & LABICHE, Y. 2015. aToucan: An Automated Framework to Derive UML Analysis Models from Use Case Models. *ACM Transactions on Software Engineering and Methodology (TOSEM),* 24, 13.

ZALKA, K. A. 2012. *Structural analysis of regular multi-storey buildings*, CRC Press.

ZAVE, P. 1997. Classification of research efforts in requirements engineering. *ACM Computing Surveys (CSUR),* 29, 315-321.

ZHANG, C., BEETZ, J. & WEISEN, M. 2015a. Interoperable validation for IFC building models using open standards. *ITcon–Journal of Information Technology in Construction,* 20, 24-39.

ZHANG, C., BEETZ, J. & WEISEN, M. 2015b. Interoperable validation for IFC building models using open standards. ITcon.

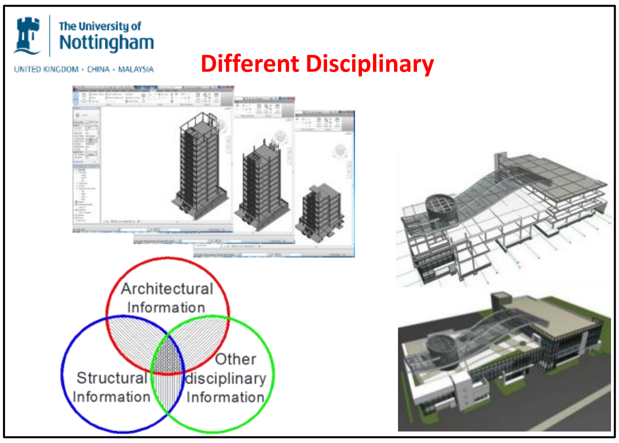
ZHAO, J. 2012. Identification and Analysis of Interoperability Gaps between Nbims/Open Standards and Building Performance Simulation Tools.

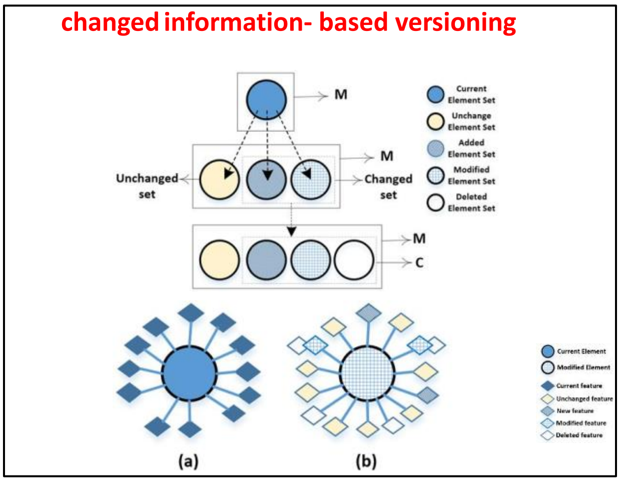
ZHILIANG, M., ZHENHUA, W., WU, S. & ZHE, L. 2011. Application and extension of the IFC standard in construction cost estimating for tendering in China. *Automation in Construction,* 20, 196-204.

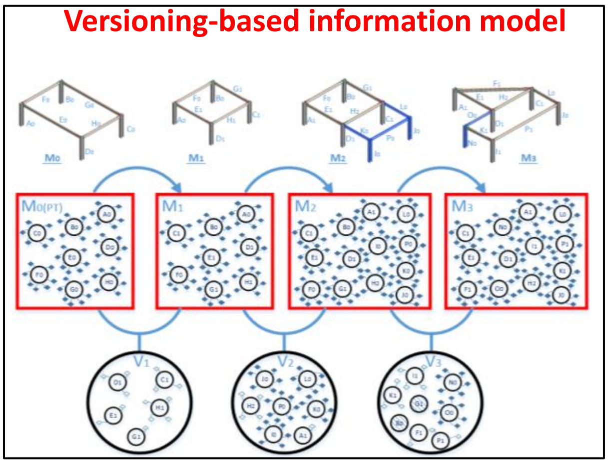
ZIMMERMAN, A. & ENG, P. 2009. Integrated design process guide. *Canada Mortgage and Housing Corporation. http://tinyurl. com/8znztzj*.

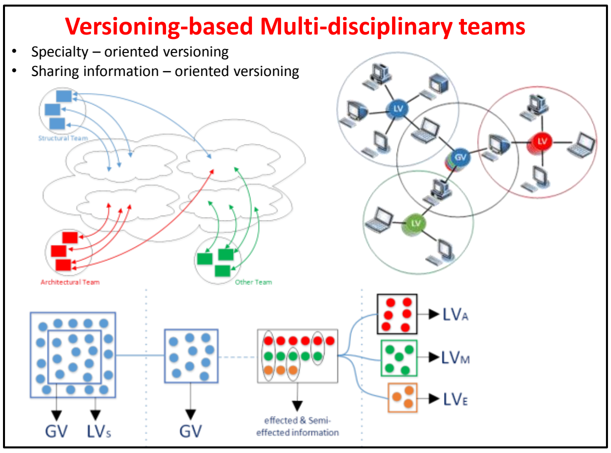
ZIMRING, C., KHAN, S., CRAIG, D., HAQ, S.-U. & GUZDIAL, M. 2001. CoOL Studio: Using simple tools to expand the discursive space of the design studio. *Automation in Construction,* 10, 675-685.

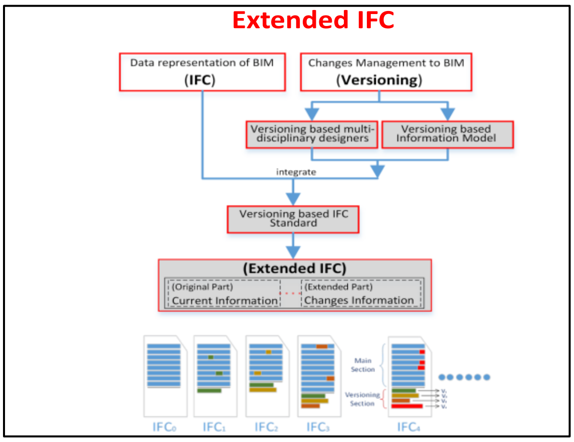
Appendix 1
Presentation Slides/ Evaluation

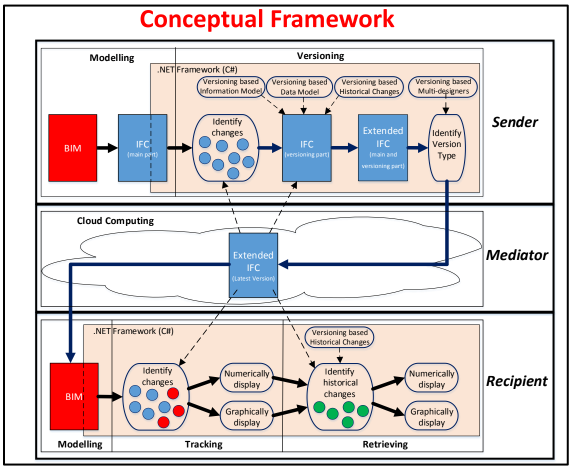

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA
Different Disciplinary
Architectural Information
Structural Information
Other disciplinary Information

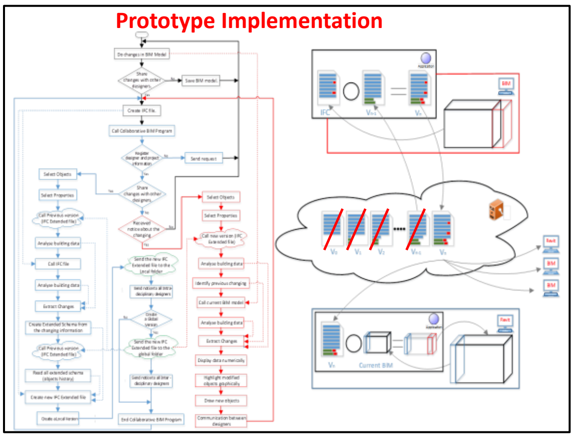

changed information- based versioning
M
Unchanged set
Changed set
M
C
Current Element Set
Unchange Element Set
Added Element Set
Modified Element Set
Deleted Element Set
(a)
(b)

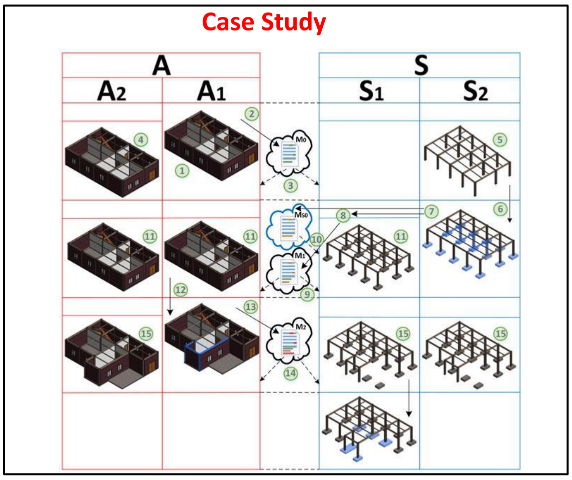

Versioning-based information model


Versioning-based Multi-disciplinary teams
• Specialty – oriented versioning
• Sharing information – oriented versioning
LV
GV
LVA
LVM
LVE
GV
LVS
GV
effected & Semi-effected information


Extended IFC
Data representation of BIM
(IFC)
Changes Management to BIM
(Versioning)
Versioning based multi-disciplinary designers
Versioning based Information Model
integrate
Versioning based IFC Standard
(Extended IFC)
(Original Part)
Current Information
(Extended Part)
Changes Information
Main Section
Versioning Section


Conceptual Framework
Modelling
Versioning
.NET Framework (C#)
Versioning based Information Model
Versioning based Data Model
Versioning based Historical Changes
Versioning based Multi-designers
Sender
BIM
IFC
Identify changes
Extended IFC
Identify Version Type
Cloud Computing
Mediator
Extended IFC
(Latest Version)
Recipient
Numerically display
Graphically display
Identify historical changes
Modelling
Tracking
Retrieving


Prototype Implementation


Case Study
A
A2
A1
S
S1
S2

Appendix 2
Evaluation Questionnaire

EVALUATION QUESTIONNAIRE

BIM - Versioning for Collaborative Design

- Please watch the PowerPoint slides and the two videos prepared before answering the questions.

Name (optional):	
Institution / Company	
Role (Designer, Contractor, Site engineer , Lecturer, Students, ...) :	
Design experience in years:	
Email (optional):	

Please rate how strongly you agree or disagree with the following statements

A. General impression	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q_1. The system allows multi-disciplinary designers to collaborate using different BIMs	☐	☐	☐	☐	☐
Q_2. The system Improves change management in BIMs	☐	☐	☐	☐	☐
Q_3. Extending IFC to deal with changes improve the collaboration among multi-disciplinary designers	☐	☐	☐	☐	☐
Q_4. The system has a positive impact on the design process in general	☐	☐	☐	☐	☐

Q_A. Which parts of the system you found that affect positively the collaboration process? and why? (modelling, versioning, sharing, tracking, retrieving)(you can choose more than one)

B. Design process	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q_5. The system displays a good degree of flexibility in operation	☐	☐	☐	☐	☐
Q_6. The system can help reduce the re-design possibilities	☐	☐	☐	☐	☐
Q_7. The system can help reduce the design process time and cost	☐	☐	☐	☐	☐
Q_8. The system improves the collaboration among different designers	☐	☐	☐	☐	☐

Q_B. What are the obstacles to the adoption of such system in the design process?

1 of 3

C. Changes in BIMs					
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q$_9$. The process of identifying design changes numerically and graphically are clear	☐	☐	☐	☐	☐
Q$_{10}$. Adding new entities into IFC schema to represent the changes in BIM are useful	☐	☐	☐	☐	☐
Q$_{11}$. Managing changes to BIM models through Collaboration-Versioning program can help reduce the design conflicts	☐	☐	☐	☐	☐
Q$_{12}$. Clarify all effective changes are essential for the designers	☐	☐	☐	☐	☐

Q$_C$. What do you think of the advantage that the sender can illustrate all the changes in the new model to all other disciplines?

D. Versioning					
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q$_{13}$. The Versioning process at the element and feature levels enable traceability of changes	☐	☐	☐	☐	☐
Q$_{14}$. Adding new entities into IFC schema to versioning the changes for different BIMs are useful	☐	☐	☐	☐	☐
Q$_{15}$. Classifying a history of each changed object improves managing the BIM model	☐	☐	☐	☐	☐
Q$_{16}$. Using the versioning concept in the IFC file to return to earlier stages of design is useful	☐	☐	☐	☐	☐

Q$_D$. Do you support the idea that versioning all changes in IFC can improve the design process?

2 of 3

E. Central Shared Storage	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q_{17}. Using the cloud-sharing service makes exchanging the shared IFC easier	☐	☐	☐	☐	☐
Q_{18}. Managing access into local and global workspaces to share the new version with the other disciplinary designers is obvious	☐	☐	☐	☐	☐
Q_{19}. Only the competent designer is authorized to contact with the other teams	☐	☐	☐	☐	☐
Q_{20}. Having a central IFC file that carrying the current model version with all change history is useful	☐	☐	☐	☐	☐

Q_E. Do you agree with the idea of having an IFC file stored in the cloud that carrying the current information with all change history?

Q_F. Any additional suggestions and further comments

Thank you

ARAS JALY-ZADA

3 of 3

End