Towards Automated Structural Stability Design of Buildings—A BIM-Based Solution

Satinder Kaur Khattra 1,*, Hardeep Singh Rai 2 and Jagbir Singh 2

1 Department of Civil Engineering, I.K.G. Punjab Technical University, Kapurthala Highway, VPO-Ibban, Distt, Kapurthala 144603, Punjab, India
2 Department of Civil Engineering, Guru Nanak Dev Engineering College, Ludhiana 141006, Punjab, India; hsrai@gndec.ac.in (H.S.R.); jsdhillion26@rediffmail.com (J.S.)
* Correspondence: satinder113@pau.edu

Abstract: Building information modelling (BIM) is a revolution in the architectural, engineering, and construction (AEC) industry. In the AEC industry, several interesting innovations are being introduced every day. Most of the code compliance work, which is time-consuming and quite multifaceted, is now performed manually. The number of unwanted errors is increasing, and any suspension of the project is led by the structural code-checking system, and is a tiresome process. Subsequently, the BIM-based model is presented to automate the structural stability design of buildings. The Industry Foundation Classes (IFC) specification is introduced in this work, and is used to automate the building design processes to achieve design correctness and efficiency. The IFC-based interoperability framework is presented to automate the design of structural elements of RC buildings. Consequently, a deep multi-perception-based generalized adaptive framework (DMPBGAF) is presented for the automated code compliance checking process. The concepts and relationships in the building code were automatically matched to their equivalent concepts. A building code was checked against the Indian seismic code IS 13920, and when shear wall data were presented in this IFC file, it exchanged the data with the custom engine. It is possible to eliminate the unwanted issues based on the automated code compliance system with BIM. This goal can be accomplished by using a coding rule-based engine to compile the BIM model and store the design code of the specific country with its clauses. After generating the Python file to code all of the prepared checks, the python file in the form of functions was obtained from the IfcOpenShell library. Finally, a rule engine was developed for interpretation of the checks; the three model IFC files were tested on this, and it gave satisfactory results. The FreeCAD software was employed in this research for the IFC-based BIM model to automate the design. The shear wall data were presented in the first IFC model; all of the code checks were passed based on this modelled design, and when given to the rule engine, it gave a 100% compliance report. Insufficiently reinforced shear wall data contained in the other two IFC files were purposely created to fail some of the code checks. This method significantly reduces the time by eliminating the duplication of efforts. From the viewpoint of accuracy and reliability, the proposed method is more accurate and results in enhanced reliability as compared to manual design methods.

Keywords: industry foundation classes; building information modelling (BIM); generalized adaptive framework; Indian seismic code; IS 13920; shear wall data; rule engine; code checks

1. Introduction

Building information modelling (BIM) is the foundation of digital transformation in the architectural, engineering, and construction (AEC) industry. Over the last few decades, digitization has been the major development in the industry, and the AEC industry has digitized nearly every stage of the building lifecycle. Across the building lifecycle, BIM technology is utilized for a vast variety of applications [1]. In the late 1990s, based on the
advent of BIM, automatic code compliance (ACC) saw an important leap [2]. Based on these regulations, building codes, and bylaws, an ACC system provides the main process for verifying the design [3]. Generalized-adaptive-framework-based techniques have been presented for ACC. The required design checks must be passed by the proposed building design so that the builders can make sure of every characteristic [4]. The design is approved after the mandatory modifications and revisions. However, a neutral format known as IFC represents a significant advancement in the BIM platform, and has contributed to improved methods of data exchange and collaboration between a variety of BIM software and the code testing system’s development. The building is described by the 3D geometric-based digital models and semantic information, to be built expansively in a BIM-based construction project [5]. The code compliance checking process for the building can be partially automated and improved using this prepared digital database [6].

A theoretical data schema is specified by the IFC, and it is used to share BIM information in the file format [7]. The EXPRESS data specification language is defined by the conceptual schema. BIM models are exchanged by the IFC standard form, and for describing elements or building systems other standard forms exist [8]. Against the compliance constraints, a construction model is checked, considering the portrayed building data [9]. Delivering semantic interoperability amongst the multifariousness of the schemata, semantic web technologies appear as a solution. BuildingSMART International (BSI) of the IFC standard’s Ontology Web Language (OWL) is used, and this affects the publication. In the semantic pre-processors and the geometrical input, the conversion algorithm’s output is linked as presented by the automating code compliance checking [10]. The chain algorithm and the pre-processors are triggered each time the semantic checker receives a new IFC model [11]. The final output is a triple store populated with data at the appropriate level of detail, and aligned with the regulations’ ontology. Additionally, this clarifies how to filter the building model data required to perform conformity control from the IFC file [12].

According to the relevant regulations, the sample building model designed within the application’s scope is shown along with the compliance check of their story height values [13]. In the Indian standards, the main seismic code is the “Criteria for Earthquake Resistant Design of Structures”, which deals with the assessment of seismic loads, and defines seismic zones and design factors. The Indian seismic code IS 13920: 2016 was meticulously exploited to create all of the needed code checks to be fed into the rule engine. Outside of application code, rules are checked and encoded in an orthogonal manner using the rule engines [14,15]. As it allows a declarative specification of rules to be maintained independently from the processes and workflow encoded in the applications, the rule engine approach is increasingly being adopted. Across the widespread application of 3D BIM design, two noteworthy obstacles hinder the AEC industry at present: With regard to the product acquisition stream, the first primary obstacle lies in risk issues related to the straightforward extraction of digital information. To execute new software solutions, the BIM is programmed by experts who regularly do not have the time required, and this is unpredictable. Most enforcement of the function of the code, which is very multifaceted and time-consuming, is now done manually. On various similar members, the majority of the inspections are performed because the manual revision of checks is very demanding. Furthermore, repeatedly applying checks on the same type of structural members results in haziness.

Consequently, the building industry, as one of the world’s traditional industries, contributes to economic development and accelerates urbanization. Nevertheless, the growth of more complex construction projects has resulted in a series of collaboration challenges for various software platforms and stakeholders, and has placed greater emphasis on cooperation throughout the whole lifespan, including construction, design, maintenance, and operation. In most situations, including data gathering and representation, data in the construction area are fragmented and dispersed across multiple sources. The success of projects is inextricably linked to how well information cooperation challenges are resolved. Subsequently, at the level of collaboration between various stakeholders, the increased
scope and complexity of construction projects poses more demands. A fundamental difficulty in the collaborative process is how to achieve greater information interoperability across numerous disciplines and technological platforms. Based on the common methods of information transmission, openBIM (building information model) can meet the needs of information interaction across different applications, and can increase cooperation efficiency and accuracy.

Although several BIM applications for the AEC industry are available on the market, most of them lack interoperability. A novel framework based on the IFC data model was developed in this study to solve interoperability concerns and to automate the design of steel reinforcement in reinforced concrete (RC) frames, as well as to promote interoperability between architectural and structural design. The design procedures were changed from a method of numerous, complex, and highly detailed drawings to the building’s single BIM model, which contains all of the needed code-checking information. Former AEC experts are extensively familiar with the design procedures and construction methods, but are unskilled with this newly developed BIM software. Section 2 portrays the literature survey; the research methodology is depicted in Section 3; Section 4 narrates the experimentation, results, and discussion; and Section 5 discloses the conclusions of this research.

2. Literature Review

This review of the literature examines the development of BIM, procedures involved in using BIM as a design tool, and the means and practices involved in checking a building design’s compliance against building codes using BIM. BIM provides the AEC industry with several additional advantages, which are missing in typical CAD drawings, thus indicating the need for this platform in the current construction sector. In the BIM industry, there is a variety of software that can be presented; this software has unique operations and different requirements of the engineer, contractor, and modeler. A common platform known as IFC is becoming popular, and can be used to export and import data into other software.

Patlakas et al. [16] presented a BIM-based framework through which ACC is delivered; this framework was introduced for the development of components. Direct or complex implementation of ACC requires a sophisticated approach; the simple categorization of the components is the main structural design problem.

İlal et al. [17] presented a recent model and an additional modeling methodology for the advancement of future ACC systems using a building code representation, wherein the four-level representation establishes a new model that combines the SMART code project’s semantic modeling approach with the theoretical foundations. The building codes’ individual rule statements, domain concepts, and relationships between rules, along with the overall organizations’ separate modeling, are allowed by the hybrid model, which organizes the representation into four levels.

An IFC-based interoperable architecture for generating linked data was presented by Hernández et al. [18] for the CARTIF-III building, where static and dynamic data were integrated in a harmonized way. A BIM-centered framework was created for this purpose, where all of the static information modelled was contained in the BIM for the building, including sensors. To integrate data from both repositories, they designed a dynamic database using the IFC paradigm and, as a result, kept track of the dynamic data that the building generates regularly.

Kincelova et al. [19] proposed a code-checking method for the National Building Code of Canada (NBCC)’s detail analysis, with this required information obtained. From the text regulation, the information is retrieved as the initial step for this method, leading to a method of architectural building model compliance checking.

Preidel et al. [20] argued that code compliance inspections are necessary because they standardize the functions that a building must eventually perform. Currently, these controls are time-consuming and error-prone, and they are largely carried out manually by planning consultants and authorized personnel. The contents of the applicable guidelines
must be converted into a digital format that machines can read, comprehend, and apply to enable automation.

Zhang et al. [21] provided a new machine-learning-based approach for automatically matching building-code ideas and relations to their Industry Foundation Classes (IFC) equivalents. For all of these systems, code–BIM matching and matching of ideas represented in the codes to those in the BIMs are required.

Manjula et al. [22] examined RC frames built to Indian specifications (IS 1893:2002 and IS 13920:1993 (2008)) and a draft proposal (IITK-GSDMA EQ 11 v 4.0, 2008) to withstand earthquakes with sufficient safety using regular constructions. By lowering the design base shear, the regular frame designs can be made more cost-effective. For normal frames, the beam/column moment–capacity ratio (M.R.) in IS 13920:2016 was determined to be uneconomical.

Jiang et al. [23] proposed a new approach to facilitate the green building assessment process, combining the benefits of the ontology and BIM. Evaluation standards included the fragmented information represented by the green building assessment, while the green building evaluation ontology (GBE ontology) for these multidisciplinary data required for the information-rich BIM served as the provider. The OWL and Semantic Web Rule Language (SWRL) rules were combined in a Jess rule engine by carrying out the reasoning process. The evaluation standard knowledge for green building reuse was achieved, and this approach can save time.

Jiang Shaohua et al. [24] suggested a global rule code-checking method for green construction. The green construction standard’s rules were classified into different rule types and converted to rules and mvdXML templates. An application program was utilized to test these rules, and in Revit a BCFier was combined with this program.

3. Proposed Research Methodology

Building information modeling (BIM) is one of the most promising recent developments in the architecture, engineering, and construction (AEC) industry. With BIM technology, an accurate virtual model of a building can be digitally constructed. This model, known as a building information model, can be used for the planning, design, construction, and operation of the facility. This helps architects, engineers, and constructors to visualize what is to be built in a simulated environment, so as to identify any potential design, construction, or operational issues. BIM represents a new paradigm within AEC—one that encourages the integration of the roles of all stakeholders in a project. The presented research aims to simplify the current complex structural code compliance system; it also utilizes IFC files. The IFC file format is used to share BIM data, and this IFC file is encouraged by the structural coding system. In the IFC file, IFC entities hold information in the form of text. All of the information about the IFC file’s structural and non-structural aspects is presented in the IFC entities, a lot of which are contained in the exported IFC files from a variety of software. In the form of a hierarchical tree, these entities are linked together, and this tree can be easily understood. Attributes are used to store information in IFC entities.

The expected investigation resulted in the development of a custom rule engine, which sped up the time-consuming process of code compliance. Due to the introduction of this rule engine, manual rework on a shear wall design was reduced to a substantial extent, resulting in significant time savings and the removal of all conceivable errors that might occur during manual code-checking operations.

In order to direct the predicted study in the desired direction, the workflow for the other necessary structural members was produced similarly, and the construction of a custom rule engine for universal use was established.

- There is essentially no technology for code-checking procedures of structural elements, hence allowing exploring in this area.
- In automated code-checking systems, there are just a few research papers in the literature that provide opportunities to work in this sector.
The current challenge in the AEC sector is that current BIM software lags in interoperability, which is a pressing issue caused by diverse software’s limited contributions to a common platform, i.e., IFC.

The abovementioned points make it obvious from past works that not much effort has been devoted to code compliance, and that only a small amount of work has been devoted to code compliance testing of building codes. Consequently, the notion of creating a rule engine verifies the building’s structural safety. Even though code checking for the structural safety of all of the buildings was time-consuming, an appropriate building element was chosen and the focus was placed on it. The shear wall is an important structural element, especially in tall buildings that can withstand significant lateral stresses. Consequently, the chosen topic—shear wall code compliance—was deemed appropriate for this work.

Figure 1 depicts the workflow diagram of this research; this presents the deep multi-perception-based generalized adaptive framework for the ACC checking process. The custom rule engine received shear wall data from the IFC model. As a result, to link structural element data to their corresponding entities in the IFC file, a detailed investigation was conducted. The detailed code of ductile procedures for RC structures was strictly prepared based on IS 13920:2016, and examined the data from the shear wall. The IfcOpenShell parsing functionality was utilized for developing a rule engine and prepared checks in Python. Shear wall data were provided to the rule engine by feeding three trials of IFC files for the interpretation of checks. A rule engine was developed using IfcOpenShell parsing functionality, and Python was used for the rule engine’s development and for coding in the prepared checks; by feeding three trials of the IFC file containing shear wall data to the rule engine, the interpretation of the checks was carried out. The rule engine delivered the result in the CSV file, which contained passed and failed checks.

3.1. Building Information Modelling

The AEC industry has been revolutionized by BIM, which is a global digital technology. Every day, new and exciting developments in the AEC (architectural, engineering, and construction) industry are launched, and several interesting innovations are being introduced. Today, BIM models enable sharing, collaboration, and versioning in ways that paper drawings cannot. All of these disciplines may be effortlessly connected via BIM collaboration with Autodesk’s BIM-360-like cloud-based technologies. Detecting any MEP, external, or internal issues before the erection or construction phases is a significant benefit of using BIM.

The BIM-enabled software packages are developed by the different vendors, aimed at different disciplines allowed by the multiplatform BIM. BIM allows users to strategize and visualize the entire project during the pre-construction phase, before the shovel hits the ground. BIM allows documentation and design to be completed concurrently, as well as allowing documentation to be easily updated in response to new information, such as site conditions. The fact that every part of the building design is coordinated is a significant benefit of the BIM project technique.

Subsequently, throughout the lifespan of the building—which depends on several checks and rules—the builders make sure that every characteristic of their proposed building design has passed the required design checks. Consequently, the design is approved after the mandatory modifications and revisions. Verifying the CC and checking the code rules against the country’s building design code, so as to identify possible faults in the design and then make the required reconsiderations, is a tedious task. In the course of the project, delays in completion are caused due to the abundance of manual work. Nonetheless, an automated BIM-based code compliance system eliminates unwanted problems. Consequently, utilizing this automated code compliance checking system reduces time and enhances reliability.
3.2. Industry Foundation Classes

The IFC specification is a standardized, digital description of the built environment, which includes civil and built infrastructure in general. For a variety of applications across a wide range of software platforms, hardware devices, and interfaces, the IFC is an open, international standard that is intended to be agnostic, vendor-neutral, and useable. IFC is described as “an open definition for BIM data that is transferred and shared among the many participants in facility management or building construction projects”, according to buildingSMART International. In building projects, employed objects are also known as building elements. Material, cost, size, name, placement, definition, etc., are the features of three-dimensional architectural elements. BIM-based applications can support IFC. In the context of a business transaction, the IFC program is frequently used to transfer data from one party to another, so as to facilitate data sharing between asset managers and AEC parties. The data may be shared in a way that encourages and facilitates interoperability, which is the advantage of using the IFC scheme. IFC allows construction companies, asset managers, engineers, and architects to share AEC data.

IFC files exported from various software programs contain a large number of IFC entities, which contain all of the information about the IFC file’s structural and non-structural elements. The entities are linked to one another in the form of a hierarchical tree, which may be understandable. Information is stored in the form of attributes in each IFC entity. BuildingSMART provides the IFC file requirements as well as an explanation of all
of the entities and their characteristics. Specifications provided in buildingSMART were utilized to understand the behavior of the IFC file, its entities, and their respective attributes. The shear wall data were compared to the information contained in IFC entities using IFC files. Consequently, IfcOpenShell—an open-source software library—was utilized for parsing the data from IFC files. The IfcOpenShell parsing functionality was explored and understood, which then helped to derive the required information from the IFC file. The language used was Python.

### 3.2.1. IFC-Based Interoperability Framework

Interconnecting the various measurement tools with BIM is one of the obstacles to overcome for the self-inspection of buildings. Due to the different formats that must be integrated into the common IFC standard, interoperability can be difficult. Because diverse platforms and software are utilized by different stakeholders throughout the project stages, in order to ensure reliable data sharing, interoperability through open formats is critical. Throughout the lifecycle of an asset, an optimized information logistics system is required for structuring and standardizing BIM data, including access to shared data and interchange file formats. Successful implementation of BIM requires planning for information sharing between tools. For technology integration and seamless interoperability, as well as to reduce data loss and maximize data accuracy, more neutral file formats must be developed.

In order to facilitate interoperability across architectural and structural design applications, the IFC data model has been used as the framework’s information model. This application is extremely interoperable, because it uses IFC as the open standard file format for BIM information sharing for both importing and exporting data. Because of the dramatically diverse representations of building parts, data transmission between architectural design and structural analysis models is particularly difficult. Data are needed to analyze the building’s facilities, and depending on the device vendor, numerous and heterogeneous data formats are accessible. Consequently, harmonization of information in a common language is required to raise the level of interoperability in system-to-system communication. In efforts to accommodate these variances, existing software tools and standards are insufficient. The geometric interpretations of data are not considered in the research on inter-domain BIM frameworks. In general data interchange frameworks, geometric interpretation analysis is typically project-specific, and is rarely reflected. Figure 2 shows a diagram of the integrated framework model.

Figure 2 shows the architectural diagram of the automated system framework, where the BIM model’s IFC file extracts the data of building beams/columns. The IFC file is also imported to the structured BIM model, and the extracted data are also utilized as shear wall geometric data. Consequently, building data, loads and material properties, shear wall geometric data, and configured files are used for the analytical model. The configured file consists of checks, general code requirements, reinforcement data, and an IS code specification table. The analytical results, shear wall data, and design code information are inserted into the automated design, which is calculated based on the developed framework. The architecture and the structural models are then updated.

In order to compute reinforcement by design codes, the framework combines functional and geometric information from architectural BIM models with design code requirements. With the use of BIM technology, 3D visualization and full automation are possible. This consists of a semantic enrichment rule engine implemented in Python programming language, integrated into an open, vendor-neutral BIM environment.
3.2.2. Structure of IFC

The BIM-based programs and IFC file format are the softwares between which the information on the building elements can be easily transferred in the models created with BIM programs. The building elements information is translated into the relevant IFC concepts when the BIM model is transformed to the IFC format. All definitions and descriptive information of IFC are regularly developed and updated by buildingSMART. In the IFC file format, the BIM model is transferred, and is required to provide automatic code compliance control for this detailed information. A BIM-based program exports the IFC data file. Likewise, based on this BIM-based program, an IFC file can be opened. With a text editor program, created IFC files can be opened and viewed. The plain texts are thus turned into total objects with type inheritance and relations defined by the IFC schema. The IFC schemata consist of three types—entities, attributes, and properties—as depicted in Figure 3. The description of this IFC scheme is explained in detail in the following subsections.
3.2.3. Entities of the IFC Format

IFC describes an entity-relationship model based on EXPRESS that contains hundreds of entities organized in an object-based hierarchy. IfcSpace, IfcBeam, and IfcBuilding are the assets that are identified by naming them after the “Ifc” prefix. In the IFC data model, the IFC entity is a uniquely identified object. Within the IFC schemata, the object is assigned default dependencies and attributes based on the entity description. BuildingSMART prepares a Data Dictionary, which describes each asset in the IFC in detail. BuildingSMART’s Data Dictionary defines 412 current assets. IfcWall is a building element entity, the IfcRelAssigns and IfcRelAggregates elements’ relation is regulated by the entity, while the IfcRelDefinesByProperties element property is contained in an asset. With a given BIM program, a three-dimensional section of an exemplary building model is created.

Table 1 shows the entities; #36 defines the wall, with attributes in parentheses, which further link to more entities to define the wall. Here, the wall defined in #36 is linked to #24 via hierarchy, so this entity—i.e., #24—contains a plan profile as a rectangular profile as well as plan dimensions, with a length of 4 units and a thickness of 0.23 units. This is how the IFC file is exploited.

Table 1. Entities of a Wall and its Hierarchy Entities.

| Entity | Description |
|--------|-------------|
| #24    | IFCRECTANGLEPROFILEDEF (. AREA., ‘rectangular’, #23,4.,0.23); |
| #25    | IFCCARTESIANPOINT ((2.,0.115,0.)); |
| #26    | IFCCARTESIANPOINT 3D (#25, #7, #6); |
| #27    | IFCCARTESIANPOINT (#21,'Body', 'SweptSolid', (#27)); |
| #33    | IFCCARTESIANPOINT (#9); |
| #34    | IFCCARTESIANPOINT (#11,'Body', 'SweptSolid', (#27)); |
| #35    | IFCCARTESIANPOINT (#5,'Structure', #33,#35,#5); |

3.2.4. Attributes of the IFC File Format

The attributes of the IFC file format are defined as data within an entity that is defined by reference to another entity. Direct attributes, inverse attributes, and derived attributes are the three types of attributes. The first four traits and definitions of each entity change depending on the entity; however, they are as follows:

- **GlobalId:** In the entire software world, a globally unique identifier is assigned.
- **Owner History:** Assigning information about the object’s present ownership, such as application, owning actor, local identity, and information about the object’s recent changes.
- **Name:** The optional name for use by the software systems or users’ participation.
• Description: In order to allow for the exchange of helpful comments utilizing an optional description.

3.2.5. IFC Model Querying

The geometry of the structure, the location of the components in a coordinate system established by the software, and the types of materials and their attributes are all included in the BIM data derived from the architectural model. The structural and non-structural member information—such as material data, member section profiles, and geometric locations—is contained in the architectural model. In terms of specific building parts—such as IfcBeam, IfcSlab, IfcColumn, and IfcWall—the structural information is usually expressed. Using a simple concrete frame, the distinctions between an architectural model and a structural model are depicted in Figure 4.

Figure 4. (a) BIM Architectural Model and (b) BIM Structural Model.

The created framework extracts data from BIM files, such as geometric data, kinds of materials, dimensions, spatial information, and other associated variables of building elements, by recursively iterating through data instances based on the rules specified in the framework. Cases of IFC files exported from different models are depicted in the following cases:

Case 1: Two separate Revit and Tekla models were associated with Navisworks using the IFC data interchange format during the refinement of a BIM model. The results show that the Coordination View 1.0 or Surface Geometry format should be chosen to preserve the model’s integrity when imported into Revit when a Tekla model exports IFC files; the Revit model is exported to *.nwc format, and the Tekla model is exported to IFC format, which can be imported into Navisworks for integration. The efficiency and refinement of the BIM model can be significantly improved by applying several hybrid modelling software approaches [25], based on the characteristics of the assembled substation.

Case 2: Apart from the customized program that reads and extracts information from IFC files exported from BIM models, this study also uses BIM-authoring tools to generate BVBS to test the program’s resilience. BIM software’s 3D design models provide better methods for finding and resolving interferences, thereby eventually reducing design and construction time [26].

3.2.6. Shear Wall Data

Lateral forces are resisted by the structural elements of a shear wall or forces that are parallel to the wall’s plane. In building construction, on a plane parallel to their planes, lateral stress is transferred from roofs, exterior walls, and floors to the ground foundation by a rigid vertical diaphragm. The reinforced concrete wall is an example. In addition
to the weight of the building and its occupants, lateral forces such as earthquake, wind, and uneven settlement loads create severe twisting (torsional) forces. Due to this, the shear leads to structural failure. In high-rise buildings subject to lateral winds and seismic stresses, shear walls are extremely critical. Shear walls are typically planar or flanged in form, whereas core walls are made up of channel sections. The former also have enough stiffness and strength to keep lateral displacements under control. The shear wall’s shape and plan position have a significant impact on the structure’s behavior. The shear walls should be placed in the center of each half of the building from a structural standpoint, but this is rarely practicable; afterwards, it makes extensive use of space. It is preferable to utilize walls with openings. As a result, the walls surrounding the stairwells and the lift shafts are frequently used, and the shear wall data in the IFC file are then loaded into the custom engine for code compliance, and the results are fetched as per the prepared code checks. Subsequently, the limitation of the proposed work is that the information extraction from the building model is often a source of error due to false, inaccurate, or absent information provided by modelers in the design phase. Nevertheless, there are several commercial ACC systems; therefore, the ACC process can be described as in the following section.

3.3. Automatic Code Compliance Checking

The ACC checking that takes place after the appearance of the IFC open data format for BIM is an important and major step. Rule control systems control and arrange the information for each element of the designed building, and the IFC file format contains detailed information. The first successful effort to automate design alignment in the construction industry started with an implementation study of decision tables representing the standard features. Studies on automated code compliance checking systems make up the major part of the research done on the topic of automatic code compliance. Users with very restricted access to the rule generation engine are considered by some of these systems. The manual regulation verification problem in a finite number of computable steps is addressed by the computational procedure, which is an automated code compliance checking system.

3.3.1. Deep Multi-Perception-Based Generalized Adaptive Framework (DMPBGAF)

Generally, four stages are considered in the ACC control, for which a DMPBGAF system is developed. The goal of this study was to create a theoretical foundation for a framework that is adaptive to the target domain and supports an open standard for converting code into computable representations. The code compliance checking module’s job is to compare the performance of the building envelope to the rules. To assist the user in checking building codes, a code compliance report is generated. In the rules package, the decision tables’ rules are read and stored. The initial stage consists of data preparation and interpreting the rules in the legal legislation, along with logical regulation of the rules in the legislation for their implementation. In the second stage, the building model is inspected to produce these data. BIM-based programs drive this data generation. The necessary rules begin to be checked at this stage. The true rule compliance verification is achieved in the third stage, when the compliance checking is performed. Reporting the compliance check results is the final step.

Rules were chosen using the implementation system’s rules engine because it provides an open-source business rules management solution that facilitates rule development. There are a large number of rules and design facts presented in the rules engine that handles the multi-object pattern-matching problem. The checking results are obtained using the rules engine, which is based on the action elements of the decision tables. To aid the building envelope design, the compliance checking findings are shown and transcribed into a report file that includes reference indices, comments, and the reference building codes. The output file of the code checks is executed in the csv file, which clearly shows the passed and failed checks.
3.3.2. Rule Interpretation Stage

The rules that were originally written in human language must be interpreted into a more appropriate format for rule checking during the rule interpretation stage. There are now two broad methods for executing this, as shown in Figure 5, where rules are either explicitly translated through the logic of human language, or translated into computer codes by a programmer. The first method is more sophisticated than the second, since it requires a high level of knowledge to define, translate, and maintain. The second method provides a means for logical interpretation that is easier to translate because it can be carried out by a construction expert without any programming background.

![Figure 5. General structure of the rule-checking process.](image)

3.4. Proposed BIM-Based Framework’s Structure

The developed DAuto application is an add-on for FreeCAD that utilizes IFC-based BIM model data for automation of the design of RC shear walls, and returns the design results to the original model with an embedded visualization tool. FreeCAD (parametric 3D modeler), IfcOpenShell (open-source IFC library and geometry engine), PySide2 (Python binding of the cross-platform GUI toolkit Qt), and Qt Designer (Qt tool for designing and building GUIs with Qt widgets) are used to develop this tool to extract data from the IFC file, while Python is used to obtain data from the user through graphical user interfaces (GUIs) and extract data from the IFC file using IfcOpenShell, and then process them to perform earthquake analysis of shear walls as per IS codes, and add the minimum required reinforcement to the shear walls. The tool for shear wall reinforcement design was established as a new module with its own workbench called “DAuto”, which is an external workbench for FreeCAD.

FreeCAD, like many other modern design applications, such as CATIA or Revit, is built on the workbench principle. A workbench is a collection of tools organized together for a certain task. The same idea applies to FreeCAD. The folder containing the source code contains—in addition the main file—an Init.py file, and sometimes an InitGui.py file. The instructions for creating a new workbench and adding new tools can be found in InitGui.py. The Init file is always executed when FreeCAD starts, and the InitGui.py file is executed immediately afterwards, but only when FreeCAD starts in GUI mode—not in console mode. This is all that is needed for FreeCAD to find a newly created workbench at start-up and add it to its interface.

3.5. Theoretical Analysis

In this research, building codes of shear walls were checked against the Indian seismic code IS 13920. IFC has resulted in improved data interchange methods, improvement in collaboration across a variety of BIM software, and code testing system improvements. The scope of work is limited to code checking of the shear wall. A custom rule engine was developed using IFC as a database for a building model and IfcOpenShell as an open-source
library to extract the data. The rule engine contained the checks of the shear wall as per the Indian seismic code.

3.5.1. IfcOpenShell for Parsing Information

A library named IfcOpenShell was used for parsing the required information from the IFC file. The key function used for parsing the data was “by_type(<argument>)”. Using the argument, the user can parse the anticipated information from the required entity. For instance, if the user wanted to obtain the data for a beam, then they would type IfcBeam as an argument, i.e., by_type (‘IfcBeam’). Correspondingly, the desired information from this particular entity can be attained utilizing the “.” operator. The procedures for drawing the required data using IfcOpenShell are shown for the IFC entities in Table 2:

Table 2. Procedures for Drawing the Required Data Using IfcOpenShell.

| IfcOpenShell Code Procedures |
|------------------------------|
| >>> # Importing the free source library to parse data from IFC file: IfcOpenShell |
| >>> import ifcopenshell |
| >>> # Opening .ifc file with help of open () function |
| >>> file = ifcopenshell. open("wall.ifc") |
| >>> # Now using by_type () for extracting ifcwall from the IFC file |
| >>> wall = file.by_type("ifcwallstandardcase") |
| >>> # Data stored in wall can viewed by typing wall |
| >>> wall |
| [#36=Ifcwallstandardcase ("3UnmqjOi2HwPta2f4VHas", #5, 'Structure', "$", #33, #35, $)] |
| >>> # Now extract all data from this entity using. Operator |
| >>> wall. GlobalId |
| '3UnmqjOi2HwPta2f4VHas' |
| >>> wall. OwnerHistory |
| #5=IfcOwnerHistory (#3, #4, $, ADDED, $, #3, #4, 1555407714) |
| >>> wall. Name |
| 'Structure' |
| >>> wall. ObjectPlacement |
| #33=IfcLocalPlacement ($, #9) |
| >>> # similarly wall representations can be stored in some name, say wall_rep |
| >>> wall_rep = wall. Representation |
| >>> wall_rep |
| #35=IfcProductDefinitionShape ($, $, (#34)) |
| >>> # Now wall dimensions can be obtained as below |
| >>> length = wall_rep. Representation. Item [0]. SweptSolidArea.XuDim |
| >>> print length |
| 4.0 |
| >>> thickness = wall_rep. Representation. Item [0]. SweptSolidArea.YuDim |
| >>> thickness |
| 0.23 |
| >>> height = wall_rep. Representations [0]. Items [0]. Depth |
| >>> height |
| 3.0 |

3.5.2. Indian Seismic Codes

During earthquakes, ground vibrations generate forces and deformations in structures. Such stresses and deformations require structures to be constructed to withstand them. Then, the structures can resist earthquake effects with minimal loss of life and property damage, because seismic codes aid in the improvement of structural behavior. In the planning, design, detailing, and construction of structures, seismic codes exist in countries all over the world to aid design engineers.
3.5.3. Preparation of Code Checks as Per IS 13920

The required code checks were strictly prepared as per IS 13920:2016—the code of ductile detailing practice of RC structures, developed under the Project on Building Codes. It is necessary to check the design assumptions before finalizing the reinforcement. Table 3 shows the necessary code checks for design provisions of a shear wall, including details such as check name, clauses referred, check formulae, etc.

Table 3. Code Checks for The Shear Wall as Per IS 13920.

| S. No | Checks                                | Formulae/Values Used | Clause   | Notations                          |
|-------|---------------------------------------|----------------------|----------|------------------------------------|
| 1     | Minimum wall thickness                 | T > 150 mm           | 9.1.2    | t = thickness of shear wall        |
| 2     | Extreme fiber stresses                 | fc_max < 0.2fck      | 9.4.1    | fc_max = extreme fiber compressive stress, fck = compressive strength of concrete |
| 3     | Number of reinforcement curtains       | Reinforcement bars should be in 2 curtains if: tv > 0.25√fck or T > 200 mm | 9.1.5    | tv = factored shear stress         |
| 4     | Minimum longitudinal reinforcement     | 0.0025t d            | 9.1.4    | d = wall length                    |
| 5     | Minimum horizontal reinforcement       | 0.0025t h            | 9.1.4    | h = height of wall                 |
| 6     | Maximum diameter of reinforcement bars | Φ < t/10             | 9.1.6    | Φ = diameter of the reinforcement bar |
| 7     | Maximum spacing of the reinforcements | Smallest of Lw/5 or 3t or 450 mm | 9.1.7    | Lw = horizontal length of the wall |
| 8     | Shear check                           | The spacing required > horizontal bars spacing | 9.2      |                                    |
| 9     | Moment check                          | Muv > Muc            | 9.3 and Annex A | Muv = total moment of resistance of the shear wall, Mu = factored bending moment acting on the wall |

3.5.4. Processing IFC Data

After all of the necessary code checks were prepared and a thorough examination of the parsing capability of IfcOpenShell was conducted, a Python file was generated to code all of the prepared checks into in the form of functions obtained from using the IfcOpenShell library. For better results, the wall parameters for length, thickness, and height were obtained as shown in Table 4:

Table 4. Wall Parameters for its Length, Thickness, and Height.

```python
wall_length = wall. Representations. Representation [0]. Items [0]. SweptArea. XDim
wall_thickness = wall. Representations. Representation [0]. Items [0]. SweptArea. YDim
wall_height = wall. Representations. Representation [0]. Items [0]. Depth
```
As seen above, for extracting the required data from the IFC file using IfcOpenShell, the parsing should be conducted through several other entities. For instance, in the above example, to obtain the wall length, the desired information has to pass through several other entities of the hierarchy of entities stored in the wall. Similarly, all of the other entities were obtained and stored in the form of functions similar to the above functions, and the required coded checks were applied, forming the required rule engine.

3.5.5. Rule Engine

A rule engine is a piece of software that helps to manage and automate rule processing by executing rules according to an algorithm. To conclude one or more actions being triggered, a rule engine integrates a set of facts entered into the system with its own rule set. The rules engine’s most essential benefit is that it separates rules from the program, and processes rules quickly. In order to change rules, the application must be more adaptive; as a result, users of the code-checking system can change the rules without the involvement of IT specialists. The rules engine is used to match these nodes, and then the matching results are obtained. The assessment report in this code compliance checking software comprises the compliance checking results, as well as the relevant design regulations, reference indices, and guidance for designers. These rules often explain the business logic that has to be executed in this environment in a declarative manner. The facts frequently change the current operational state of the system. Different conflict resolution strategies can be used by a rule engine to decide the order in which conflict rules are executed. There are two execution techniques in a rule engine:

- **Forward Chaining:** This is a “data-driven” approach used to extract new facts using available facts and inference rules, until a goal is attained when facts are inserted or the rule engine is changed, at which point one or more matching rules will be true at the same time and schedule for execution. As a result, the rule engine begins with facts and is completed with a conclusion.

- **Backward Chaining:** In contrast to forward chaining, this is a “goal-driven” or inference approach that is reversed. Backward chaining begins with a conclusion or a list of objectives that the engine attempts to satisfy. If it is unable to meet these objectives, it searches for sub-objectives that will assist it to meet some of the present objectives. The process is repeated until the initial conclusion is proven.

4. Experimentation and Result Discussion

In a BIM context, this framework presents an automatic code generation process at the conceptual level. During the early stages of structural design, by decreasing the iteration process, this framework has the potential to help engineers by saving them time and effort. A modest construction project was modelled with FreeCAD to analyze an IFC file in detail. The 2 × 3 schema version of the sample building model developed with FreeCAD was exported with an “.ifc” extension. The sample buildings and sampled shear walls were presented. The structure of the building consisted of several stories and rooms, and the 3D view, 3D sectional view, and floor plans of the building are shown in Figure 4. In order to create the designed and fully reinforced models of shear walls, prominent open-source BIM software was used. The models thus-prepared were exported in an IFC file to achieve the objectives of the research. Initially, only a single wall model was developed, and then the number of shear walls was increased. The single-wall model in FreeCAD and IFC Viewer is described in Figure 6.
Figure 6. Wall model in FreeCAD and IFC Viewer.

Figure 6 shows the wall model in FreeCAD and IFC Viewer. FreeCAD is a piece of free and open-source general-purpose parametric building information modeling software with finite element method support and a 3D computer-aided design modeler. IfcOpenShell is another piece of software utilized to conduct the IFC export and import in FreeCAD. From FreeCAD, it can export to IFC. IFC Viewer also allows sharing of the BIM models online through a web browser, without needing the installation of any application to view the work. IFC Viewer provides a toolbar of options for working with the model. X-ray mode makes the walls of the model transparent. Normally, a shear wall contains a certain amount of reinforcement in both the horizontal and vertical directions. Due to the increases in height of the structure, the effects of wind forces increase in reinforced concrete-framed structures. Sway or horizontal movement is restricted by codes of practice.

4.1. Installing Workbench

Through the add-on manager, which is available via FreeCAD’s graphical user interface, it is easy to install newly created workbenches in FreeCAD. Alternatively, the folder containing the source code can be placed in the MOD directory of FreeCAD through Macro. Code was added to the Python path so that after restarting FreeCAD it appeared in the BIM workbench, among other tools, as shown in Figure 7.

Figure 7. DAuto installed in FreeCAD’s API.

4.2. Graphical User Interface of the Tool

The user interface is responsible for the communication between the designer and the system. With a GUI, the visual display itself becomes a source of user input. It shows
various graphical objects in the form of icons and input devices, such as buttons and scroll bars. The user can interact directly with these objects by selecting a button or scrolling a scroll bar. Through its GUI, DAuto is supposed to be more adaptable, extendable, and simple to use, as represented in Figure 8.

Figure 8. Graphical user interface of the tool.

This project used Qt Designer for creating the GUI. In order to design and build GUIs with Qt Widgets using the Qt tool called Qt Designer, DAuto provides a complete solution for pre/post-process requirements, and enables the users to implement their design in an integrated platform. In particular, most of the changeable requirements have already been considered and solved by the configured files. DAuto has a well-designed, adaptable, and easy-to-maintain framework enabled by the object-oriented GUI method.

4.3. Implementation of the Tool

The EXPRESS language, which is a modelling language rather than a programming language, was used to create the exported IFC model. To enable the production of structural models, information in the IFC model about building items and their attributes must be transferred to a programming language. The Python programming language was used in this research. Python is unable to read IFC files. By implementing IfcOpenShell into the tool, it was possible to read the IFC files. Technical visualization of the model could be achieved by opening the FreeCAD model using IfcOpenShell in the program. IfcOpenShell uses the Python library to query and parse IFC files in order to achieve placement exactly as expressed in the IFC schema. IfcOpenShell supports all IFC2x3 and IFC4 entities.

4.4. Working of the Tool

When the user starts FreeCAD, the start page containing a group of workbenches is displayed. Subsequently, when clicking the button for the DAuto workbench, a window pops up containing a tool for the design of steel reinforcement for the shear wall (Figure 9). There are two important buttons in the interface: the first is “Choose”, used to import the IFC model to the tool database, and the second is “Run Analysis”, used to execute the tool. In the pop-up windows, the GUI for the workflow of the program allows the user to select IFC files that are stored locally on the device.
When the user loads an IFC file, by selecting the “Choose” button on the top right-hand side of the GUI, the file is imported into the GUI. Although the complete workflow normally runs without further user interaction, if required, the user may edit some of the parameters displayed in the GUI. Then, after selecting the “Run Analysis” button, the tool is executed to read the model information and extract objects’ information to enable the visualization of the textual results.

The data provided by the pre-processing stage are utilized by implementing a group of functions for calculating the analysis results of a structural model. Subsequently, various structural inputs—such as load cases, steel sections, and support conditions—can be assigned automatically to create a structural analytical model. Once the results of the structural analysis have been computed in the processing stage, the data post-processing becomes responsible for making these results available to users by graphically displaying or printing them. The “Add Reinforcement” button on this panel is an option to show the updated original BIM model.

4.5. Modules and Library that Carry Out the Process

For the analysis of structure, the module for computing load was established. Dead load is computed with the geometric information and the material information extracted from building elements. Live load is configured with the general information and the design code information. Various design codes are required for the formation and type of structures. In this study, the library of design codes was established through configuration files. The source code in the present study was configured with IS codes. Nevertheless, it can be extended according to the design code of any country with little effort, and without changing the source code. This type of configuration makes the tool much more flexible. The limit state checks are specified based on IS codes. According to IS 13920, all structural limit states were verified in this study.

4.6. Design of the Shear Wall

To illustrate the proposed methodology, it was applied to the automated design of RC shear walls. BIM models were prepared using BIM authoring software and saved as IFC models in the system. The required steel reinforcement was then calculated for each shear wall, considering the stored bending moment and shear force results based on the minimum required steel reinforcement, in addition to other conditions stipulated in the IS design codes. The entire computational process was driven automatically through code developed in Python using the application programming interface (API) of FreeCAD. Python scripts can completely control FreeCAD, and using Python scripts provides an API for solid modeling.

The bending model’s distribution for shear wall data for structural model 1, structural model 2, structural model 3, and structural model 4 is depicted in Figure 10. The original
BIM model could be updated automatically with the steel reinforcement design results, and all of the shear walls were reinforced accordingly, thereby facilitating design communication and the generation of detailed construction drawings, while also determining the clashes, if any.

Figure 10. (a) Structural Model 1, (b) Structural Model 2, (c) Structural Model 3, and (d) Structural Model 4.

In addition to columns, slabs, and beams, RC buildings often have vertical plate-like RC walls called shear walls. These walls usually begin at the foundation level and continue throughout the height of the building. The spacing between vertical reinforcements can vary, as the force borne by the shear wall at the ends is much more than at the middle part of the wall. Accordingly, the spacing between reinforcing bars is lesser at the ends of the walls than at the center part. The shear wall samples were presented for the code compliance checking process. The sample shear wall data are represented in the following figure.

The testing of the rule engine in Figure 11 shows the shear wall models. After this, the models of the shear wall along with the buildings were developed. The building contained all the basic structural and non-structural components, such as beams, columns, staircases, slabs, walls, windows, and doors, along with the shear walls, as shown. Several models in their modelling and post-modelling state are shown.

The header and the body are the two components that make up the basic structure of an IFC file. The header section contains information such as the IFC version used, the creation date, etc., belonging to the building model. When the IFC file’s header section for the sample building model presented in the table is inspected, it is clear that the IFC file’s scheme is in the IFC 2 × 3 format. The geometry and attributes of the building model are described in the body section. This section also contains details on how the relationships between each piece of the model were established.
Entity #9 in the IFC file presented in Table 5 has properties that are associated with entities #6, #7, and #8, which define #9. These entities are linked together in the form of a hierarchical tree, which is simple to comprehend. Attributes are used to store information in IFC entities. As a result, a thorough investigation was conducted to link structural element data to their corresponding entities in the IFC file. The hash symbol (#) is the beginning of each line of data in the IFC file’s body section. Starting with # 1 is the numbered data lines. Each numbered line contains an IFC object. The IFC file for the sample building model presented in the table has 13 data lines. The number of data lines in an IFC file grows as the number of building elements (columns, beams, walls, floors, doors, windows, roofs, stairs, etc.) in the model grows, as does the preference for material qualities of the elements and each program procedure. Each element of the building model and each activity for the design of the building are represented in the BIM application (FreeCAD).

Table 5. Basic Form of IFC File.

| ISO-10303-21; |
| HEADERS; |
| FILE_DESCRIPTIONS (('View Definitions [Coordination Views]',)';); |
| FILE_NAME('wall\[msn\]', '2019-0b-12T12:33:45';) |
| FILE_REL{'SCHEMA(('IFC2×3'));}; |
| ENDSBEC; |
| DATA; |
| #1 = IFCPERSON ($) _, $, $, $, $, $; |
| #2 = IFCORGANIZATION ($) _, $, $, $; |
| #3 = IFCPERSONANDORGANIZATION (#1, #2, $); |
| #4 = IFCAPPLICATION (#2, '0.18 build 16093 (Git)', 'FreeCAD', '118df2cf_ed21_438e_a41'); |
| #5 = IFCONونيERHISTORY (#3, #4, $, ADDED, $, #3, #4, 1557664425); |
| #9 = IFCDIRECTION ((0.9998,0.000012,0.)); |
| #10 = IFCDIRECTION ((0.0000000165,0,0.999987)); |
| #11 = IFUCARTESIANPOINT ((1.0,0.,0.)); |
| #12 = IFCAxisPlacement2D (#8, #7, #6); |
| #13 = IFUCDIRECTIONS ((0.,1.,0.)); |
4.7. Demonstration and Verification of the Tool

A prototype building model was developed to demonstrate the functionality of the proposed tool for bidirectional flow between architects and structural engineers. Two additional building models were created to verify the applicability of the novel data-exchange framework.

To demonstrate the proposed framework, three case studies were developed. The model was created in FreeCAD and, using the default settings of the software, the model was exported as an IFC model. Consequently, after starting the automated structural design tool, through the “Choose” button in this tool, the IFC-based architectural model was imported and the structural building elements and their corresponding geometric information, section profiles, and material information were extracted.

After selecting the “Run Analysis” button, the tool analyzes the model, and a new window pops up to show the details of reinforcement provided in the shear wall, as shown in Figure 12.

![Figure 12. Shear wall design reinforcement.](image)

After this stage is executed, the “Add Reinforcement” button sends the designed reinforcement back to update the original model, as depicted in Figure 13.

![Figure 13. Design reinforcement in the shear wall.](image)

The optimal steel reinforcement design results are then visualized in 3D using a BIM model in IFC, enabling design communication and the development of detailed construction drawings.

The IfcRelAggregates entity’s definition and properties are summarized as follows:
• This entity aids in the collection of objects.
• It has six characteristics (between two parentheses, the parts separated by commas),
• The fifth attribute specifies where the elements will be collected.
• The sixth attribute specifies which elements will be gathered.

These data can be simply exported from the IFC file using a programming language based on these examinations. As a result, the model data can be obtained, which is the most crucial stage in automatic code compliance checking.

4.8. Code Compliance Checking

The following steps should be taken to compare the regulations received from the legislation with a project created in a BIM-based program, and to determine whether the project complies with the legislation.

• In the relevant legislation, rules are determined.
• The BIM application exports the IFC file for the drawn project.
• The IFC file’s stated rule is defined by the hierarchy of the parameters.
• The script is written so that the data can be exported from the IFC file based on the provided hierarchy.
• If mathematical procedures on the data retrieved from the IFC file are required, they are carried out.
• The outcomes are compared to the regulations’ rules.
• The message appears in the file that has been complied, and the report of the results is generated.

The creation of the sample building model and checking of its working capability are mandatory after the development of the custom rule engine, which can be achieved by taking several trials of different IFC files and applying the following procedure:

• First, an IFC model is created, which is then fed into the custom rule engine. The intended shear wall data are used to apply checks on this file.
• The file is compiled by the rule engine using IfcOpenShell’s parsing functionality.
• After that, the custom rule engine gives the result output file, which shows all of the passed and failed checks.
• The findings are supplied in a CSV file that can easily distinguish between passed and failed inspections.

Taking advantage of the BIM platform of interoperability, and using a common platform format for different software—namely, IFC—we adopted a firm methodology for developing the rule-checking engine for shear wall data from an IFC file. The methodology used a, open-source library called IfcOpenShell, which is very useful in parsing the necessary data from the required IFC file and adopting an unsurpassed language named Python for coding the necessary checks in the file, concluding the development of the engine. The results obtained after completing all of the mentioned objectives provided a satisfactory amount of data.

• After resolutely studying the structural entities of various IFC files containing shear wall data, the necessary entities obtained for extracting the required information from the IFC file included IfcWallStandardCase, IfcReinforcingBar, and their respective hierarchy entities. The IFC entities named IfcRelContainedInSpatialStructure and IfcRelAggregates failed to perform their functionality, i.e., to assign or link the reinforcement bars to their respective parent members.
• However, entities such as IfcShapeRepresentation, IfcExtrudedAreaSolid, IfcRectangleProfileDef, and IfcAxis2Placement3D help to obtain the geometric properties of various members and assign reinforcing bars to their parent members.
• A thorough study of the IFC entity named IfcCartesianPoint obtained from IfcPolyLoop provided coordinates for various types of reinforcement bars, which were used for finding spacing between reinforcement bars and differentiating between vertical
and horizontal bars. IfcMaterial and IfcRelAssociatesMaterial were useful entities for obtaining the material assigned to the shear wall and reinforcement bars.

- The code IS 13920’s clause 9 was carefully read, and the required checks were obtained for application to the shear wall. The required manipulations were carried out to convert the code checks to rule engine feeding checks. The second objective was fulfilled as seen in Table 4 in the previous chapter.
- After the rule engine was developed using the parsing functionality of IfcOpenShell and coding of the prepared checks using Python, interpretation of the checks was performed by feeding three trials of the IFC file containing shear wall data to the rule engine.

The command window for the compiled file is depicted in Figure 14; the first IFC file contained fully designed shear wall data satisfying all of the checks as per IS 13920. To compile this file, first, open the command windows and go to the directory where the rule engine file is located. The Python command is used to open the rule engine. It will then ask to wait until the rule engine loads. The rule engine takes a few seconds to load because of the libraries used for generating the rule engine. Pandas, NumPy, and Maths, in addition to IfcOpenShell, are common libraries used in generating the rule engine. Each of the libraries has its own functionality. After a few seconds, it will ask for a file to be compiled. Consequently, the file name is input without extension. Once the required file is input, the message will appear showing that the file has compiled and the report of the results has been generated.

Table 6 provides the output file generated by applying checks using the rule engine on a first file containing appropriately designed shear wall data, which was designed manually to pass all of the checks. The other two IFC files were intentionally created with shear wall data having failed several checks, so as to observe the compiling capability of the rule engine.

The second file contained shear wall data the design of which fails in terms of both shear and minimum reinforcement in both directions. The output obtained after feeding the second IFC to the rule engine is represented in Table 7. Subsequently, after the complete development of the rule engine, similar shear wall data files were used as inputs for obtaining the results to check the adequacy of the engine.
Table 6. Report File Generated for Shear Wall 1.

| S.No | Report for Wall:1                                                                 |
|------|----------------------------------------------------------------------------------|
| 1    | Thickness of every part of wall is more than 150 mm: Check passed by 80.0 mm    |
| 2    | Wall thickness is sufficient as per extreme fibre stresses                        |
| 3    | Reinforcements should be provided in two curtains: check passed                  |
| 4    | Minimum longitudinal reinforcement: Check passed by 35.58%                       |
| 5    | Minimum transverse reinforcement: Check passed by 7.76%                         |
| 6    | Maximum diameter of reinforcement: Check passed by 11.0 mm                       |
| 7    | Maximum spacing of vertical reinforcement: Check passed by 160.0 mm              |
| 8    | Maximum spacing of horizontal reinforcement: Check passed by 85.12 mm            |
| 9    | Shear check passed by: 153.81 Kn                                                |
| 10   | Moment check passed by: 4140.13 Kn-m                                            |

// ~ Compliance results for shear wall: 100% checks are passed ~ //</!

Table 7. Report File Generated for Shear Wall 2.

| S.No | Report for Wall:2                                                                 |
|------|----------------------------------------------------------------------------------|
| 1    | Thickness of every part of wall is more than 150 mm: Check passed by 80.0 mm    |
| 2    | Wall thickness is sufficient as per extreme fibre stresses                        |
| 3    | Reinforcements should be provided in two curtains: check passed                  |
| 4    | Minimum longitudinal reinforcement: Check failed by 39.36%                      |
| 5    | Minimum transverse reinforcement: Check failed by 66.13%                        |
| 6    | Maximum diameter of reinforcement: Check passed by 15.0 mm                      |
| 7    | Maximum spacing of vertical reinforcement: Check passed by 161.82 mm             |
| 8    | Maximum spacing of horizontal reinforcement: Check failed by 65.86 mm            |
| 9    | Shear check failed by: 37.05 Kn                                                 |
| 10   | Moment check passed by: 2985.65 Kn-m                                            |

// ~ Compliance results for shear wall: 60% checks are passed ~ //</!

The third IFC file contained shear wall data in which the shear wall was badly designed and modelled to fail most of the checks, including shear failure, moment failure, and the failure of minimum reinforcements in both directions. Table 8 represents the output generated from the third file. Although the suggested work for this research was accomplished, there were plenty of problems faced during the progression of the research. The IFC database provided by buildingSMART has a great richness, but most of the software fails in transferring the important data to the IFC file when exported. The geometric properties and IFC entities providing coordinates, member placement, member orientation, and member extrusion direction were used to obtain the data that an IFC file might have provided, but failed to perform such a task.
Table 8. Report File Generated for Shear Wall 3.

| S.No | Report for Wall:3 |
|------|-------------------|
| 1    | Thickness of every part of wall is more than 150 mm: Check passed by 80.0 mm |
| 2    | Wall thickness is sufficient as per extreme fibre stresses |
| 3    | Reinforcements should be provided in two curtains: check passed |
| 4    | Minimum longitudinal reinforcement: Check failed by 39.36% |
| 5    | Minimum transverse reinforcement: Check failed by 66.13% |
| 6    | Maximum diameter of reinforcement: Check passed by 15.0 mm |
| 7    | Maximum spacing of vertical reinforcement: Check passed by 161.82 mm |
| 8    | Maximum spacing of horizontal reinforcement: Check failed by 65.86 mm |
| 9    | Shear check failed by: 50.72 Kn |
| 10   | Moment check passed by: 286.48 Kn-m |

// ~ Compliance results for shear wall: 40% checks are passed ~ //

Figure 15 portrays the success rate of the code checks. The number of code checks and their success rate are given. The figure shows that when the number of codes or missions increases, the success rate also increases; therefore, the proposed method gives the best performance. Consequently, when the code reaches 200, the success rate is 62%, and when the number of missions reaches 1000, the success rate is approximately 98%.

Figure 18. Success rate for the passed code checks.

Figure 19 illustrates the percentage of automated structural design achieved for shear wall, slab, beam, and column data. The automation in structural design and analysis, the automation and BIM in the structural design and analysis, and generative designs and BIM in the structural design and analysis are disclosed in this figure. The figure shows that the automation checks for slabs achieved a better percentage than for the beams and columns.

Figure 16 illustrates the percentage of automated structural design achieved for shear wall, slab, beam, and column data. The automation in structural design and analysis, the automation and BIM in the structural design and analysis, and generative designs and BIM in the structural design and analysis are disclosed in this figure. The figure shows that the automation checks for slabs achieved a better percentage than for the beams and columns.
Figure 18. Success rate for the passed code checks.

Figure 19 illustrates the percentage of automated structural design achieved for shear wall, slab, beam, and column data. The automation in structural design and analysis, the automation and BIM in the structural design and analysis, and generative designs and BIM in the structural design and analysis are disclosed in this figure. The figure shows that the automation checks for slabs achieved a better percentage than for the beams and columns.

Figure 19. Percentage of automated structural design achieved.

Figure 20 depicts the average run time of the research, which is compared with the other methods. The average run time for BIM-based automated modeling was compared with existing methods, such as multi-agent reinforcement learning (MARL) systems. Consequently, comparing this average run time with the existing method, the proposed method requires less run time and, therefore, improves the system performance.

Figure 20. Performance graph of average run time.

4.9. DAuto Validation

The validation of the proposed DAuto tool was carried out by comparing the steel reinforcement obtained from manual calculation with that obtained using the tool. For this, a building frame of G + 9 floors was considered, consisting of four bays in both directions. The spacing along the X and Y directions was 6.0 m, and the story height was taken as 3.0 m. The building was located in Delhi.

The design parameters considered for manual design calculations are shown in Table 9, in which the beam size and the column size are represented as 600 mm × 300 mm, and
their loads are 4 kN/m² and 2 kN/m², respectively. The story height is 3.0 m and its load is 3 kN/m²; the unit weight of concrete is 25 kN/m³ and its load is 1.5 kN/m². The compressive strength of the concrete is M-25, its load is 5700 kN, and the steel grade is F-415.

Table 9. Materials used in the proposed work.

| Dimensions and Materials                  | Loads          |
|-------------------------------------------|----------------|
| Beam size 600 mm × 300 mm                 | 4 kN/m²        |
| Column size 600 mm × 300 mm               | 2 kN/m²        |
| Story height 3.0 m                        | 3 kN/m²        |
| Unit weight of concrete 25 kN/m³          | 1.5 kN/m²      |
| Compressive strength of concrete M-25     | 5700 kN        |
| Steel grade F-415                         |                |

It can be observed that the results provided for steel reinforcement calculations by the developed tool are the same as those obtained from manual calculations. A comparison of steel reinforcement results is provided in Table 10. From the perspective of time efficiency, the suggested framework is more efficient than the manual design calculations, as the manual design relies on the expertise and experience of the structural engineer.

Table 10. Comparison of reinforcement design.

| Method                              | DAuto Tool                   | Manual                      |
|-------------------------------------|------------------------------|-----------------------------|
| Vertical reinforcement in the middle| 10 mm diameter @300 mm c/c   | 10 mm diameter @300 mm c/c  |
| 0.8 × length of the wall            |                              |                             |
| Vertical reinforcement in the end    | Nine 20 mm diameter bars     | Nine 20 mm diameter bars    |
| strips                              |                              |                             |
| Horizontal reinforcement            | 10 mm diameter @292 mm c/c   | 10 mm diameter @292 mm c/c  |

5. Conclusions

Building information modelling is the most efficient platform for exchanging information for construction, design, and facility management. BIM is a common requirement for interoperability. In the form of IFC entities within the IFC file format, BIM-based programs are created from the information of the obtained building element models. The advantage of using IFC as a common platform for a variety of software can be exploited for the purposes of the anticipated research. Various IFC files containing shear wall data were studied carefully, keeping in mind the specifications provided by buildingSMART. The desired IFC entities were carefully exploited by the structural and geometric properties of shear wall data contained in the respective IFC files. After the complete development of the rule engine, similar shear wall data files were used as inputs for obtaining the results to check the adequacy of the engine. The conclusions of this research after obtaining the results provided valuable information:

- Structural IFC entities such as IfcWallStandardCase and IfcReinforcingBar, along with their hierarchy entities, provided satisfactory data that can be used in the development of the rule engine. However, entities required for associating the reinforcement bars with their parent structural members viz. IfcRelAggregates and IfcRelContainedInSpatialStructure failed to execute their tasks.
- Therefore, these entities of IFC files are not yet capable of using code compliance works. The entities IFC concrete material properties and IFC steel mechanical material properties also failed to fetch data, which as per buildingSMART should provide the material properties. The use of IfcMaterial entities provided the stasis factory data for materials associated with structural members.
• The collaboration of prepared checks with IfcOpenShell worked out excellently for the development of the rule engine.

• The three model IFC files tested on the final developed rule engine for interpretation of the checks gave satisfactory results. The first IFC model containing shear wall data gave a 100% compliance report when inputted to the rule engine.

• The other two IFC files containing shear wall data with insufficient reinforcement were intentionally generated to fail some of the code checks. On inputting these two files into the rule engine, the resultant report generated the same results as those of the manual procedure.

The present research was intended to automate the design of structural elements of RC buildings using integrated frameworks, and our results can also be used for code-checking procedures of a single coherent structural element, such as a shear wall. However, our findings can also be further explored for other structural elements such as beams and masonry constructions, i.e., the research can be used as a basis for the development of a general-purpose rule engine through the collaboration of all such assets, which can be guided for a complete code compliance report of the structural model data. The automated code compliance technique based on the BIM with IFC and IS 13920 leads to inordinate time saving and numerous error reductions.

Author Contributions: Conceptualization, S.K.K. and H.S.R.; methodology, S.K.K.; software, S.K.K.; validation, S.K.K., H.S.R. and J.S.; formal analysis, S.K.K.; investigation, S.K.K.; resources, S.K.K.; data curation, S.K.K.; writing—original draft preparation, S.K.K.; writing—review and editing, S.K.K.; visualization, S.K.K.; supervision, S.K.K.; project administration, S.K.K.; funding acquisition, H.S.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research has not received external funding.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Acknowledgments: The authors would like to mention and acknowledge the Department of Research, Innovation, and Consultancy of I.K. Gujral Punjab Technical University, Kapurthala Road, Jalandhar, for giving them this opportunity and providing them with all of the required assistance to write this research paper. This study is a part of a Ph.D. research project carried out at I.K. Gujral Punjab Technical University, Jalandhar (Punjab).

Conflicts of Interest: The authors declare no conflict of interest.

References
1. Bloch, T.; Sacks, R. Clustering Information Types for Semantic Enrichment of Building Information Models to Support Automated Code Compliance Checking. J. Comput. Civ. Eng. 2020, 346, 04020040. [CrossRef]
2. Nawari, N.O. A Generalized Adaptive Framework GAF for Automating Code Compliance Checking. Buildings 2019, 94, 86. [CrossRef]
3. Jiang, S.; Wu, Z.; Zhang, B.; Cha, H.S. Combined MvdXML and semantic technologies for green construction code checking. Appl. Sci. 2019, 97, 1463. [CrossRef]
4. Porto, M.F.; Franco, J.R.Q.; Vianna, B.F.; Porto, R.M.A.B. Automatic Code Checking Applied to Fire Fighting and Panic Projects in a BIM environment-BIMSCIP. Syst. Cybern. Inform. 2017, 15, 86–90.
5. Jamal, K.A.A.; Mohammad, M.F.; Hashim, N. Building Information Modelling (BIM) for Sustainable Industry, the Malaysian Architect’s Perspective. Alam Cipta 2019, 12, 61–72.
6. Sydora, C.; Stroufia, E. Rule-based compliance checking and generative design for building interiors using BIM. Autom. Constr. 2020, 120, 103368. [CrossRef]
7. Fan, S.L.; Chi, H.L.; Pan, P.Q. Rule checking Interface development between building information model and end-user. Autom. Constr. 2019, 105, 102842. [CrossRef]
8. Anderson, A. Using Building Information Modeling to Transform the Building Codes Compliance Process. In Construction Research Congress 2020, Computer Applications; American Society of Civil Engineers: Reston, VA, USA, 2020.
9. Dimyadi, J.; Amor, R. BIM-based Compliance Audit Requirements for Building Consent Processing. In Proceedings of EC-PPM 2018: eWork and eBusiness in Architecture, Engineering and Construction; CRC Press: Boca Raton, FL, USA, 2018; pp. 465–471.

10. Bus, N.; Roxin, A.; Piccinbono, G.; Fahad, M. Towards French Smart Building Code, Compliance Checking Based on Semantic Rules. *arXiv* 2019, arXiv:1910.00334.

11. Mastino, C.C.; Baccoli, R.; Frattolillo, A.; Marini, M.; Di Bella, A. *Noise from Plants Systems and Building Information Modeling, the Code Checking*; Universitätsbibliothek der RWTH Aachen: Aachen, Germany, 2019.

12. Fahad, M.; Bus, N.; Fies, B. Semantic BIM reasoner for the verification of IFC Models. In eWork and eBusiness in Architecture, Engineering and Construction; CRC Press: Boca Raton, FL, USA, 2018; pp. 361–368.

13. Temel, B.; Basaga, H. Investigation of IFC file format for BIM-based automated code compliance checking. *J. Constr. Eng.* 2020, 136, 113–130. [CrossRef]

14. Beach, T.H.; Hippolyte, J.L.; Rezgui, Y. Towards the adoption of automated regulatory compliance checking in the built environment. *Autom. Constr.* 2020, 118, 103285. [CrossRef]

15. Ajayi, S.O.; Oyebiyi, E.; Alaka, H.A. Facilitating compliance with BIM ISO 19650 naming convention through automation. *J. Eng. Des. Technol.* 2021, 1. [CrossRef]

16. Patlakas, P.; Livingstone, A.; Hairstans, R.; Neighbour, G. Automatic code compliance with multi-dimensional data fitting in a BIM context. *Adv. Eng. Inform.* 2018, 38, 216–231. [CrossRef]

17. Ilal, S.M.; Günaydın, H.M. Computer representation of building codes for automated compliance checking. *Autom. Constr.* 2017, 82, 43–58. [CrossRef]

18. Hernández, J.L.; Lerones, P.M.; Álvarez, S.; Bonsma, P.; van Delft, A.; Deighton, R.; Braun, J.D. An IFC-based interoperable framework for building linked-data. In Proceedings of the 6th Linked Data in Architecture and Construction Workshop (LDAC) CEUR Workshop Proceedings, London, UK, 19–21 June 2018; pp. 44–51.

19. Kincelova, K.; Boton, C.; Blanchet, P.; Dagenais, C. Fire safety in tall timber building, A BIM-based automated code-checking approach. *Buildings* 2020, 10, 121. [CrossRef]

20. Preidel, C.; Borrmann, A. Refinement of the visual code checking language for an automated checking of building information models regarding applicable regulations. *Comput. Civ. Eng.* 2017, 1, 157–165.

21. Zhang, R.; El-Gohary, N. A machine-learning approach for semantic matching of building codes and building information models BIMs for supporting automated code checking. In *International Congress and Exhibition Sustainable Civil Infrastructures*; Springer: Cham, Switzerland, 2019.

22. Manjula, N.K.; Nagarajan, P.; Pillai, M. Performance Evaluation of RC Buildings Designed as per Indian Seismic Codes, A Study on Frames with Vertical Geometric Irregularity. *J. Inst. Eng. India Ser. A* 2018, 99, 677–687. [CrossRef]

23. Jiang, S.; Wang, N.; Wu, J. Combining BIM and ontology to facilitate intelligent green building evaluation. *J. Comput. Civ. Eng.* 2018, 325, 04018039. [CrossRef]

24. Jiang, S.; Wu, Z. A BIM-based code checking approach for green construction. *ICCREM 2017*, 1, 156–163.

25. Xu, Y.; Zhang, J.; Li, D.; Ao, C. BIM Model Integration of Concrete and Steel Structures in Assembled Substations. *Adv. Comput. Sci. Res.* 2019, 91, 65–70.

26. Liu, Y.; Li, M.; Wong, B.C.; Chan, C.M.; Cheng, J.C.; Gan, V.J. BIM-BVBS integration with openBIM standards for automatic prefabrication of steel reinforcement. *Autom. Constr.* 2021, 125, 103654. [CrossRef]