



Universidade do Minho
Escola de Engenharia

**Adoption of Open-Source BIM Authoring in
Architectural Design Process**
Nigar Ibrahimzade

Nigar Ibrahimzade
**Adoption of Open-Source BIM Authoring in
Architectural Design Process**

BIM A+ European Master in
Building Information Modelling

The European Master in Building Information Modelling is a joint initiative of:



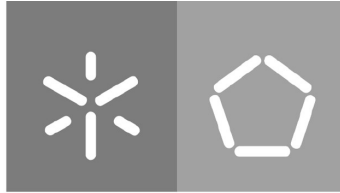
Univerza v Ljubljani



UMinho | 2021



October 2021



Universidade do Minho

Escola de Engenharia

Nigar Ibrahimzade

**Adoption of Open-Source BIM Authoring in
Architectural Design Process**



European Master in
Building Information Modelling

Master Dissertation

European Master in Building Information Modelling

Work conducted under supervision of:

Jose Luis Duarte Granja

Bruno Acacio Ferreira De Figueiredo

Rui Dias (tutor in company)



Co-funded by the
Erasmus+ Programme
of the European Union

October 2021

AUTHORSHIP RIGHTS AND CONDITIONS OF USE OF THE WORK BY THIRD PARTIES

This is an academic work that can be used by third parties, as long as internationally accepted rules and good practices are respected, particularly in what concerns to author rights and related matters.

Therefore, the present work may be used according to the terms of the license shown below.

If the user needs permission to make use of this work in conditions that are not part of the licensing mentioned below, he/she should contact the author through the RepositóriUM platform of the University of Minho.

License granted to the users of this work



Attribution

CC BY

<https://creativecommons.org/licenses/by/4.0/>

ACKNOWLEDGEMENTS

Firstly, I would like to extend my gratitude to the European Commission for supporting me financially throughout my studies and giving me a chance to get such an important diploma.

I would like to offer my warmest wishes to my main supervisor, José Granja, who supported me throughout the process that eventually led to the completion of this dissertation.

Big thanks to Bruno Figueredo, for giving me advice and supporting me throughout this journey.

Thank you, Mr. Rui Dias, my tutor in the company, for giving me an opportunity to work with you and tutoring me on this topic that I was new at.

I would like to thank my family and friends for being there when I needed them the most. Especially, in these days of pandemy, when living and coming up with something valuable is so hard for so many reasons.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

A handwritten signature in black ink, appearing to be 'Hugo', written in a cursive style.

RESUMO

Devido ao aumento da escala dos projectos de arquitectura, à complexidade dos processos de construção, e ao aumento da implementação do BIM, há uma maior procura de software de autoria profissional de modelação BIM.

No entanto, num mundo onde o software proprietário domina o mercado, nem sempre é possível alcançar uma interoperabilidade total dos dados com software proprietário. Além disso, como as empresas de software controlam estritamente a direcção do desenvolvimento de software, a indústria torna-se dependente dos seus produtos e políticas financeiras. Esta situação cria problemas para alguns campos da indústria de AEC (Arquitectura, Engenharia e Construção) e causa dificuldades para a interoperabilidade entre o software utilizado. Por outro lado, OpenBIM, como abordagem padrão de intercâmbio de dados, pode satisfazer bem as necessidades de interacção de informação entre diferentes softwares e melhorar a eficiência e precisão da colaboração, ao mesmo tempo que satisfaz as necessidades de todas as partes interessadas. No entanto, para implementar plenamente o conceito OpenBIM, a indústria terá de implementar mais software de fonte aberta (OSS). Actualmente, o termo open source tem sido cada vez mais implementado em projectos arquitectónicos, no entanto, há pouca investigação sobre o potencial do OSS para substituir totalmente as actuais ferramentas de autoria proprietárias do BIM.

O objectivo desta dissertação é compreender se o software BIM de código aberto pode suportar totalmente todo o ciclo de vida do projecto e analisar os pontos fortes e fracos do processo. De acordo com a investigação realizada até agora, não existe actualmente nenhum levantamento satisfatório do potencial total do Software BIM Aberto utilizado no contexto da indústria AEC (Arquitectura, Engenharia e Construção). Esta dissertação visa preencher a lacuna e introduzir uma revisão analítica da utilização de Software BIM Aberto de acordo com as tarefas de Projeto, a competência organizacional, a documentação da empresa de projeto, e a compatibilidade. Neste documento, são introduzidas e investigadas as normas relacionadas com o BIM aberto, possível utilização de software, normas para a interoperabilidade da informação, e capacidade de concepção com aplicações, com base em websites e literatura relacionados. Espera-se que os resultados da investigação beneficiem a comunidade de projetoarquitectónico, fazendo avançar a sua compreensão dos factores que afectam a aceitação da tecnologia do Software Open BIM e o seu impacto no desempenho de uma empresa de projeto. Além disso, a análise sistemática da teoria e prática do software de código aberto neste artigo pode apoiar a sua investigação e aplicação futuras.

Palavras chave: (Código aberto, OpenBIM, Modelação BIM, software de código aberto, FreeCAD, BlenderBIMadd-on, LibreCAD, TAD designer)

ABSTRACT

Due to the increase in the scale of the architectural projects, the complexity of construction processes, and the increase in BIM implementation, there is a higher demand for professional BIM authoring software.

However, in a world where proprietary software dominates the market, not always full interoperability of data can be achieved with proprietary software. Furthermore, as software companies strictly control the direction of software development, the industry becomes dependent on their products and financial policies. This situation creates problems for some fields of the AEC (Architecture, Engineering & Construction) industry and causes difficulties for the interoperability among the used software. On the other hand, OpenBIM, as a standard approach of data exchange, can meet the needs of information interaction among different software and improve the efficiency and accuracy of collaboration while meeting the needs of all stakeholders. However, to fully implement the OpenBIM concept, the industry will need to implement more open-source software (OSS). Nowadays, the term open source has increasingly been implemented in architectural designs, yet there is little research on the OSS potential to replace fully current proprietary BIM Authoring tools.

The purpose of this dissertation is to understand if Open Source BIM software can fully support the project's whole life cycle and analyze the strong and weak points of the process. According to the investigation done till now, there is currently no satisfactory survey of the full potential of Open BIM Software used in the context of the AEC (Architecture, Engineering & Construction) industry. This dissertation aims to fill the gap and introduce an analytical review of Open BIM software use according to the Design tasks, the organizational competence, and documentation of the design firm, and compatibility. In this paper, the Open BIM-related standards, possible software use, standards for information interoperability, and designer capability with applications are introduced and investigated based on related websites and literature. The research results are expected to benefit the architectural design community by advancing their understanding of the factors that affect the acceptance of Open BIM Software technology and their impact on a design firm's performance. Furthermore, the systematic analysis of the theory and practice of Open Source software in this paper can support its further research and application.

Keywords: (Open source, OpenBIM, BIM authoring, Open-Source Software, FreeCAD, BlenderBIM add-on, LibreCAD, TAD designer)

TABLE OF CONTENTS

1.	INTRODUCTION	11
1.1.	RESEARCH METHOD	11
1.1.1.	Data source	11
1.1.2.	Definition of evaluation dimensions	12
1.2.	DISSERTATION STRUCTURE	13
2.	OPEN-SOURCE SOFTWARE	15
2.1.	OVERVIEW AND DEFINITION	15
2.1.1.	History	16
2.2.	FREE SOFTWARE AND OPEN SOURCE SOFTWARE	17
2.3.	PROGRAMMING LANGUAGE	18
2.4.	LICENSES	19
2.5.	SOFTWARE DOCUMENTATION	21
2.6.	OPEN SOURCE MOVEMENT	23
2.6.1.	Open Source Architecture	25
2.7.	DIFFERENCE BETWEEN OPENBIM AND OSS	27
2.8.	ADVANTAGE AND DISADVANTAGE OF OSS	29
3.	ANALYSIS OF OPEN-SOURCE SOFTWARE FOR ARCHITECTURAL DESIGN	31
3.1.	OSS IN ARCHITECTURAL DESIGN	31
3.2.	BIM SOFTWARE EXPECTATION	34
3.3.	DETAILED ANALYSIS OF OPEN-SOURCE APPLICATIONS	34
3.3.1.	FreeCAD	34
3.3.2.	LibreCAD	54
3.3.3.	TAD Designer	57
3.3.4.	BlenderBIM add-on	59
3.4.	SUMMARY OF SOFTWARE ANALYSIS	62
4.	SURVEY ANALYSIS	65
4.1.	COMPARISON OF PROPRIETARY AND OPEN-SOURCE SOFTWARE	65
4.1.1.	First Case: LibreCAD and AutoCAD®	65
4.1.2.	Second Case: FreeCAD and Revit®	66
4.1.3.	Third Case: BlenderBIM add-on and Revit®	69
4.2.	SURVEY ABOUT OPEN-SOURCE SOFTWARE WITH PROFESSIONALS IN AEC SECTOR	71
5.	CONCLUSIONS	76
	BIBLIOGRAPHY	78

LIST OF FIGURES

Figure 1 - The open source software license relationship.....	21
Figure 2 - Open source software documentation configuration.....	22
Figure 3 - Structure of a typical .FCStd file.....	37
Figure 4 - Standard FreeCAD interface in 0.19.....	39
Figure 5 - Customize menu.....	40
Figure 6 -A spreadsheet with certain cells filled with text and quantities.....	49
Figure 7 - Arch Material properties.....	51
Figure 8 - Example .FCMat file.....	52
Figure 9 - Multi-Material wall element and property settings.....	52
Figure 10 - Interface of LibreCAD.....	56
Figure 11 - User interface elements.....	59
Figure 12 - Blender and BlenderBIM interface.....	60
Figure 13- IFC classification mistake in Revit®	70
Figure 14 - Chart showing BIM awareness among AEC companies and Professionals.....	72
Figure 15 - Chart show software choice among AEC Professionals.....	73
Figure 16 - Chart shows OSS awareness among AEC Professionals.....	74
Figure 17-Software criteria preferability chart.....	75

LIST OF TABLES

Table 1: Open-source software in the Architecture and design field.....	32-33
Table 2: FreeCAD import and export file formats.....	37-38
Table 3: FreeCAD workbench list.....	41-42
Table 4: List of external workbenches for FreeCAD.....	43-46
Table 5: Draft workbench File Formats.....	48
Table 6: Plug-in List for LibreCAD.....	56-57
Table 7: Summary of Software analysis.....	62-63
Table 8: Summary of comparison between chosen applications.....	71
Table 9: Survey questions about BIM awareness in AEC companies.....	72
Table 10: Survey questions about Software used by AEC professionals.....	73
Table 11: Survey questions about OSS awareness among AEC professionals.....	74

1. INTRODUCTION

The primary function of Building Information Modelling (BIM) is to allow all stakeholders involved in a construction project to communicate and cooperate at different levels and depths and to complete the insertion, extraction, and update of the project information. All parties can achieve accurate and timely information sharing through various software or platforms. Nowadays mainstream BIM software comes from large software vendors, such as Autodesk[®], Bentley[®], Trimble[®]. Although the BIM software developed by various software vendors has its own strengths, the degree of mutual interoperability is still low. The same company's software can be very compatible, while the information interoperability between the software of different companies is very low. Unable to share information with other participants resulting in information loss and information traffic, which will increase the cost of construction projects and delay the construction period (He, 2011). Moreover, the software of prominent software vendors is an integral product and cannot be modularized and decomposed (Heavey & Dagkakis, 2015). Even though some software has interfaces for developing plug-ins, these still cannot meet the needs of the construction industry to customize BIM software functions.

Faced with the problems of inability to customize, modularize, and high software prices, users have been seeking solutions, and open-source software is a better attempt. The so-called open-source software, in simple terms, refers to the open-source code users can freely use, copy, modify and redistribute software according to a specific open-source agreement. This approach of freely sharing software and software source code is fundamental to the development of software. Many open-source software has active communities, and people discuss and explore the versatility and scalability of the software online through the Internet. There is also open-source software in the BIM field, but overall it is still in the process of exploration. This dissertation objectively demonstrates BIM open source software's characteristics, functional focus, and deficiencies through data analysis. It aims to solve the following problems of open-source software: definition of open-source and open-source related movements; the professional field of open-source software application; the computer language used by open-source software; the open-source license used by BIM open source software; what methods are used by BIM open-source software to support their development and online community maintenance; what are the most commonly used document types for BIM open source software; detailed analysis of open-source software's features and finally a comparison of open-source software with proprietary software.

1.1. Research method

1.1.1. Data source

In order to deeply understand and study the problems of BIM open-source software, the first step is to determine an appropriate boundary for the research content.

This dissertation only studies the software that is used for the design stage of the project. There are many design-related software included in the Design and BIM field, such as; Parametric design

software, geometric modelling software, sustainable analysis software, structural analysis software, visualization software, deepening design software, model comprehensive collision check, cost management software, operation management software (Taher, 2016).

This dissertation only studies open source software and software with open source licenses. Some software or application users can view the code but do not include a clear open-source license; this software will not be within the scope of this dissertation (exception only those who plan to transfer to open-source). This research first conducts a comprehensive literature search. After understanding open-source and how it relates to architecture, the second step is to select the open-source software to analyze, and last, compare them to non-open-source applications. The primary method is to screen and verify through open-source websites. This article first uses the Google search engine to search, in addition to searching for some well-known open-source software publishing platforms, including:

- 1-Github: <https://github.com/>;
- 2-Google:<https://code.google.com/archive/>;
- 3-Source Forge: <https://sourceforge.net/>;
- 4-CodePlex: <http://www.codeplex.com/>.
- 5-OSArch: <https://wiki.osarch.org/>
- 6-FreeCAD wiki: https://wiki.freecadweb.org/Main_Page

The review presented in this dissertation eliminates software that does not clearly indicate that it is an open-source project and is not in the design field and then eliminates those that are inactive and have not been updated for a long time. This dissertation can determine the active status of software by checking the continuity of the submission record of each software.

1.1.2. Definition of evaluation dimensions

Since there are no mature BIM or design-oriented open source software evaluation methods, this dissertation draws on the evaluation dimension of open-source software in other fields. For example, Logothetis and Stylianidis (2016) proposed the evaluation dimensions of open source software, which mainly include: content management, content collection, metadata management, retrieval support, interoperability, and other 12 aspects. Dagkakis and Heavey (2015) provided a more comprehensive evaluation dimension of open source software in researching open-source software discrete event simulation software, which mainly includes computer language, open-source license, latest update time, domain, and version control. Logothetis and Stylianidis (2016) also proposed relevant BIM software evaluation dimensions in the study of BIM open source software and cultural heritage digital documents. Based on the evaluation methods of open source software in other fields and combined with the research content of this article, this dissertation constructs the evaluation dimensions of BIM open source software suitable for this dissertation:

(1) Computer language. Refers to the computer language used by BIM open source software. Each language has different characteristics. For example, C++ is static, while scripting languages are dynamic. This will result in C++ having better geometric data processing performance, and a dynamic scripting language can improve development efficiency.

(2) Open source license. Different open-source licenses have different license contents. For example, the GPL license (General Public License) is currently the most widely used open source software license in the open-source software industry, providing legal permission to copy, distribute, and modify software. Other licenses such as BSD (Berkeley Software Distribution), MIT (Massachusetts Institute of Technology), AFL (Academic Free License) and Apache have their own agreement requirements.

(3) The latest update time. Refers to the time when the software was last updated, to analyze whether the software is still being maintained and updated.

(4) Feature type. Since the research object of this article is open source software for architectural design process, it is necessary to understand what open source software can offer to support the project. For example, some software has the visual direction of 3D modelling and the direction of data integration and BIM standard management. Other software specializes in 2D Drawing and construction documentation, while the third can provide tools for data export from the model, like QTO or Scheduling. It is essential to understand the limits of open-source software for the design stage of creating the project.

(5) User documentation. There are many different open-source software documentation forms, including text documents, online documents, and video tutorials.

(6) Communication method. The rapid development of open-source software benefits from free and extensive exchanges and discussions. Software developers and users communicate through mailing lists, online forums, and some common communication platforms.

1.2. Dissertation Structure

Besides Introduction, the dissertation structure consists of four parts. The structure starts with Chapter 2, "Open Source Software," which clarifies the open-source concepts, the creation history, effect on historical movement, the structure, and benefits or disadvantages. Additionally, it shows the connection to the BIM, how it affected one of the leading digital movements in AEC (Architecture, Engineering and Construction).

The next chapter, "Analysis of Open Source Software for Architectural Design," is divided into four subchapters. First, section 3.1 explains the role of OSS in Architectural design and shows existing open-source applications to support architectural design projects. The shown applications are divided into workflow categories and evaluated by four essential characteristics for open source software: programming language, update date, license, and documentation. The following section 3.2 explains BIM expectations from open-source applications and what should be considered when developing the software. Section 3.3 going further in analyzing open-source software for architectural design. Four chosen applications (FreeCAD, LibreCAD, BlenderBIM add-on, TAD Design) were evaluated accordingly to capability to support the design project. The last section 3.4 ends chapter 3 with a conclusion showing a summarized analytical table of evaluated software.

Chapter 4, "Survey analysis," consists of two sections. The first section 4.1 illustrates a survey among AEC field professionals about BIM and OSS knowledge and use. The survey was done around 45

people using to understand how much open-source applications are known and how much they are in use compared to proprietary software. The last section 4.2 dedicates the analytical comparison of chosen open-source software with proprietary applications (LibreCAD versus AutoCAD®, BlenderBIM add-on, and FreeCAD, versus Revit®) to understand if open-source programs can replace applications of dominating private companies. Finally, the chapter shows OSS's strong and weak points and how much they can support the design process.

Lastly, the conclusion summarizes the result of the study.

2. OPEN-SOURCE SOFTWARE

For more than 20 years, the Open Source Initiative (OSI) has raised awareness and embraced open-source software, and built bridges between open source communities of practice. As a global non-profit organization, OSI advocates for software freedom in society through education, collaboration, infrastructure, and managing the Open Source Definition (OSD), and preventing abuse of the ideals and ethics inherent in the open-source movement (Nicholson, 2018).

Open-source software is created by many people and communities and distributed under an OSD compliant license which grants all rights to use, study, modify and share the software with modified and unmodified form. Therefore, software freedom is essential to enable the community development of open-source software (Ankerholz, 2020).

Free software has long remained confidential in businesses and administrations, but Open Source has been increasingly widely distributed in recent years. As a result, open-source solutions are now a credible alternative to proprietary software, even for complex applications. Next, the definition of Open Source and its benefits for users will be discussed.

2.1. Overview and Definition

The name was created by the Open Source Initiative, which defines the conditions that an open-source license must meet. Born in 1998 out of the Free Software Foundation, the Open Source Initiative is a non-profit organization dedicated to promoting open-source software. The two names are almost equivalent, the main idea being that software should be free and accessible to everyone. They nevertheless represent different schools of thought and philosophy. The Free Software Foundation has a vision that could be described as more dogmatic, emphasizing ethics, while the Open Source Initiative has a more pragmatic, more business-oriented approach. They consider that the most important criterion is quality for software to spread massively, particularly in companies. To compete with proprietary software, it is not enough that the solutions are free and open. They must also be at least as efficient as their competitors (Perens, 1998).

Software is considered Open-Source when its source code is freely accessible, modifiable, studiable, and transferable. Therefore, it must respect four fundamental freedoms to be deemed free and Open-Source: execution of the program study and adaptation to its needs, redistribution improvements, and dissemination of improvements. The official definition of the Open Source Initiative is based on 10 points, which can be viewed on its site (Opensource.org, 2007).

Open Source solutions are not free of rights. The person or company who wrote the program owns the copyright. It is free to use its software and fixes the license to use it. Open Source software is therefore under an Open Source license. This guarantees their Open Source character and gives them the right to use, modify, and redistribute them freely. There are two leading families of Open Source licenses. The BSD license is the most "liberal." It allows any use of a program, code, and derivative works, including integration with proprietary software, which is not under an open-source license. The GNU GPL license authorizes distribution and modification, provided that it is under the same license. The

variant of this license adapted to SaaS mode is the AGPL license. Indeed, a GPL license allows you to take an Open Source program, modify it and not repost its sources if it is for internal use or if you offer cloud access. With the advent of SaaS software, the AGPL license was created to make source code distribution mandatory even for a cloud program. If one subscribes to an Open Source SaaS application under the AGPL license, they will be able to access the sources and download them locally to make changes.

2.1.1. History

In the 1950s and early 1960s, the first free programs were developed by Tech firms. They were distributed as a free addition in a package that included the main component-hardware that actually cost money. In addition, the first free programs included source code which made it possible to edit them, fix their issues or add features if necessary. Academic circles were among the first adopters of open-source technology and their principles to facilitate the free flow of information made it easy to develop the programs further.

In the late 1960s, however, software development became a separate business. And consequently, it became more and more distant from the equipment that they were bundled with and turned into a separate branch of business for tech firms. Therefore, by that time, lawyers started creating restrictive licenses. ARPANET (Advanced Research Projects Agency Network) researchers used RFCs (Research Fund for Coal and Steel) to develop telecommunication protocols. Ultimately, the Internet appeared in 1969.

In the 1970s, AT&T (an American multinational conglomerate holding company) introduced the first versions of UNIX (family of multitasking, multi-user computer operating systems). They were free, but it was impossible to modify their source code. In the late 1970s and early 1980s, paid software gained more traction among software companies and computer vendors. However, legally they were protected by the government and its judicial body (Bretthauer, 2001).

In the 1980s, the software became popular through BBS (bulletin board system) systems. Software written in BASIC (Beginner's All-purpose Symbolic Instruction Code) and other languages were distributed as source code only. The number of free software has grown significantly since the 1980s. Richard Stallman started the GNU (GNU's Not UNIX) Project and created the Free Software Foundation. This is how the first companies were created, for which free software was the main business.

In the early 1990s, the first free software with the Linus Torvald kernel was introduced to the public. It was combined with the GNU operating system. Debian, which was founded by Ian Murdoch in 1993, also adhered to the principles of free software GNU and FSF (Free Software Foundation). The Linux deal began in the late 1990s. This is how the site-based companies emerged. They used free servers a lot, especially the Apache HTTP (HyperText Transfer Protocol) server. The LAMP stack (Linux, Apache, MySQL, PHP) became more and more popular. The Freeware Summit, hosted by Tim O'Reilly, brought together all the free and open systems leaders. Through a developer vote, open source was defined as a new term over original software (Gonzalez-Barahona, 2021).

In the early 2000s, large software corporations realized that free software could be a serious competitor to their core business. Microsoft's Steve Ballmer named Linux "cancer," referring to a copyleft license. Other companies have claimed their intellectual property rights (Greene, 2001).

But in recent years, the movement in open source projects has been growing. Many well-known companies are adapting to free software. New business models are emerging. The emergence of popular web services and their open-source use was perceived as financial madness in the past. Today it is already the new norm. Many government agencies have adopted free software and open innovation as fundamental building blocks in building a successful information society.

The US government has developed the National Open Government Action Plan, which includes an open data policy with the goal of "making information resources accessible, discoverable, and useful to the public to help stimulate entrepreneurship, innovation, and scientific discovery — to improve the quality of life for Americans" lives and makes a significant contribution to job creation" (Government, 2019).

2.2. Free Software and Open Source Software

We often think that open source software is necessarily free when it is first and foremost open. In theory, a publisher could charge for the distribution of its Open Source ERP. In practice, the community version of open source software is often free because charging for software that someone else might offer for free doesn't make sense. Open Source software is therefore necessarily free, but the service is chargeable: company version, integration, assistance, maintenance, training, additional developments. Indeed, the best Open Source business software is contrary to popular belief of equal quality to proprietary software, and it is expected that some services are paid or that there may be a "business" version integrating services and maintenance. Be wary of "bogus Open Source," where the free and community version is unpatched, full of bugs, and with only part of the functionality.

Open source doesn't just mean access to source code (Stallman, 2015). The conditions for distributing free software must meet the following criteria:

1. Free redistribution

The license does not prevent any party from selling or giving the software away as part of the global software distribution that contains programs from several sources. The license will not require a royalty or other fee for such a sale.

2. Source Code

The program must include source code and allow distribution in source code form and compiled form. When a product is not distributed with the source code, there must be a well-known way to obtain the source code at the most reasonable reproduction cost, preferably by downloading it free from the Internet. Source code should be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a pre-processor or translator are not allowed.

3. Derivative works

The license must allow modifications and derivative works and distribute them under the same conditions as the original software license.

4. Integrity of the author's source code

The license may restrict the distribution of the source code in modified form only if the license permits the distribution of "patch files" with the source code to modify the program at build time. The license must explicitly authorize the distribution of software created from modified source code. The license may require that derivative works have a name or version number different from the original software.

5. No discrimination against individuals or groups

The license must not discriminate against a person or a group of people.

6. No discrimination against fields of activity

The license must not prevent anyone from using the program in a specific field of activity. For example, it cannot restrict the program from being used in business or genetic research.

7. License distribution

The rights attached to the program shall apply to all whom the program is redistributed without the need to execute an additional license by such parties.

8. The license must not be specific to a product.

The rights attached to the program should not depend on whether the program is part of particular software distribution. If the program is extracted from this distribution and used or distributed under the program license terms, all parties to whom the program is redistributed should have the same rights as those granted with the original distribution of the software.

9. The license must not restrict other software.

The license must not impose any restrictions on other software distributed with the licensed software. Thus, for example, the license should not insist that all other programs distributed on the same medium must be open-source software.

10. The license must be technologically neutral

No provision of the license may be based on any individual technology or interface style (Initiative, 2007).

2.3. Programming Language

This section discusses the definition and detailed explanation of programming language and open-source programming terms in more detail. Until now, researchers and scientists have described several statements for the mentioned scientific area, which can be found below:

1. A programming language is a set of instructions for writing programs and software, which are specifications of a computation or algorithm (Aaby, 2004).
2. A computer programming language is a type of language which is used to develop computer programs, which involves a machine to perform some kind of calculation and computation (Languages, 2015).
3. Open source programming is a computer science process that is deliverable by a specific license or agreement that lets users modify, redistribute, study, or progress (Randhawa, 2018).

The programming process includes main steps such as: analyzing the problem, writing the pseudo code, reading the data generating codes and algorithms, testing the accuracy of projects and realizing and developing applications using the selected programming language. In order to keep the applications and software up to date with novel methods and technologies, most of the libraries and

applications need to spend money for steps such as installation, training, updating the code, and so on. Open source programming is the ideal solution for cost-efficient and comfortable coding. Using Open-source programming, developers can quickly code and modify the algorithm to adjust the program for their needs. It is accessible to the general public, and everybody can use it. There are many language options for open source programming so that developers can choose a language and develop their projects. Each language has its distinct libraries and features, which sets them apart from each other. Python is a powerful and portable open-source programming language that can be used as a standalone program as well as for developing applications. Python can be used for database management, manipulating the data, and developing the project. The main Python libraries that are used are listed below:

- Requests: Using this HTTP library, the process of creation of requests to python application is quite simplified. Requests also offer JSON methods that automated the addition of strings to URL
- NumPy: The tasks related to the scientific computation of this work are done using one of the powerful libraries of python, NumPy.
- Pandas: Another library that is used for developing mathematical formulas and algorithms is Pandas. It helped solve problems with different types of data structures, reshaping, manipulating, and merging datasets.
- FastApi: FastAPI is a high-performance web framework similar to Django, and Flask is used for building APIs.
- Tkinter: For creating a graphical user interface, the Tkinter library is used for this project. It has an essential collection of vital visual widgets (Costa, 2020).

Despite the mentioned libraries, other libraries such as Matplotlib, Plotly, Pyodbc, Scipy have also been implemented for visualization, plotting data, and technical computation, accessing databases.

2.4. Licenses

Open source software licenses regulate the usage, modification and distribution of software code. The permission that is needed to use, share or edit the code for applications is given in the form of licenses.

More than 80 types of licenses exist, and they typically belong to 2 different subgroups:

- Copyleft licenses
- Permissive licenses

Copyleft license is one in which code developed from open source code inherits the provisions of the original license. Permissive license, on the other hand, provide the user with more freedom to edit, distribute and reuse applications

The most used copyleft licenses are GPL, EPL and Mozilla

The General Public License (GPL) is relevant for commercial, patent, and private usage, and it keeps licensing notices and copyright terms. Any software that incorporates GPL code must make all of its source code available under the same license. If GPL code is utilized and distributed in a work, all of the source code must be released under the same GPL license. The GPL is a strong copyleft license because of this restriction (Berman, 2020).

The Lesser General Public License (LGPL) is called the sister license of the GPL license. They both are the leaders in the value and philosophy of free software, and are the most fundamental driving force for the development of open source software. While being similar to each other, the biggest difference between them is that smaller projects or objects accessed through bigger licensed projects do not require the larger project to be distributed. Furthermore, the updated source does not need to be distributed under the same conditions as the rest of the code project (Berman, 2020).

The Eclipse Public License (EPL) is predominantly used by developers of business applications the Eclipse Public License (EPL) is often utilized for business-oriented applications. Software written in EPL, non-EPL code, and even proprietary code can all be merged and sublicensed using EPL, as long as non-EPL portions are kept separate as modules or objects. Under the EPL license, adjustments can be implemented, but they must be provided under the same terms.

The Mozilla Public License (MPL) is the license that gives user the most freedom among copyleft open-source licenses. They make it simple to alter and utilize their code in closed-source and/or proprietary applications, as long as any MPL code is preserved in separate files and delivered with the package. The MPL also governs patent grants and the retention of copyright notices.

The most used permissive open source licenses are Apache, MIT, and Unlicense.

According to **Apache License**, license notifications and copyrights must be included in the disseminated code and/or as a notice in the software. Derivative works, larger projects, and modifications, on the other hand, are free to have various licensing terms and are not needed to disclose source code when disseminated. A patent grant is included in Apache licenses.

MIT License, named after the renowned university where it was developed, is probably the most commonly applied open source license in the world, owing to its conciseness, clarity, and simplicity. It let anyone to do whatever they want with the original code as long as the copyright and license notice are included in the distributed source code or software. It exempts writers of any liability and does not contain a patent grant (Goldstein, 2021).

GNU Affero General Public License (AGPL) is a type of copy-left license that gives permission to every developer to render, adjust or distribute the work as long as any output copies have links to the same copy-left licensing scheme. One of the main advantages of AGPL is the possibility to keep the adjusted part of code snippet written by the open source community. At the same time, AGPL prevents other users from selling open-source software (FOSSA, 2021).

Unlicense: As the name implies, this is the most permissive of open source licenses, as it effectively makes the open source public domain. There are no restrictions, which mean that these unlicensed works can be distributed without source code and under a variety of situations (Berman, 2020).

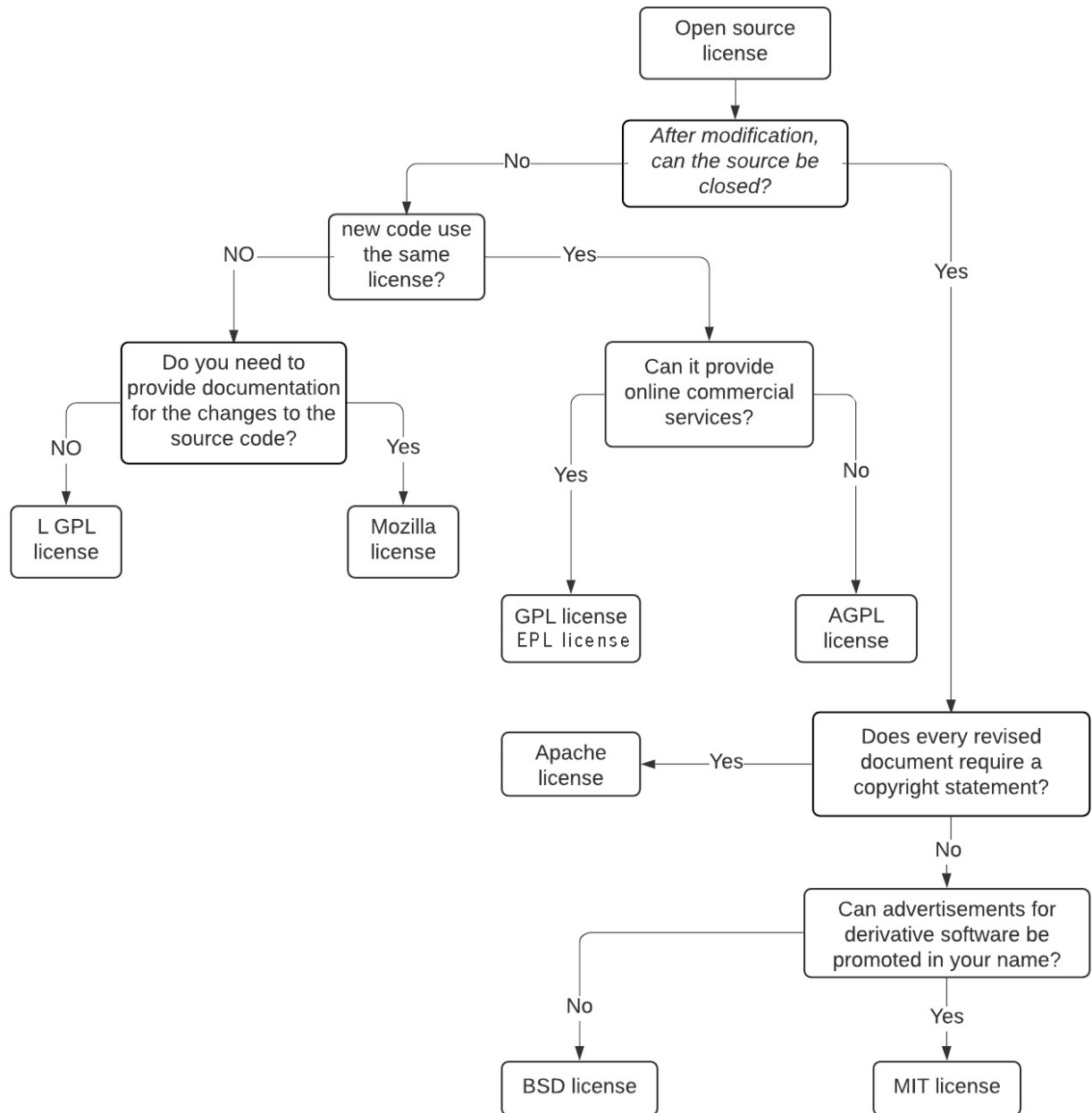


Figure 1: The open source software license relationship. Created by author.

2.5. Software Documentation

Effective interaction with the application and the eventual positive feedback from the user start from quality documentation. According to a survey organized by Github in 2017, most of the complaints regarding open-source programs are about their seemingly confusing documentation (Davies, 2018). A

dependable and usable application almost always requires carefully made and informative documentation.

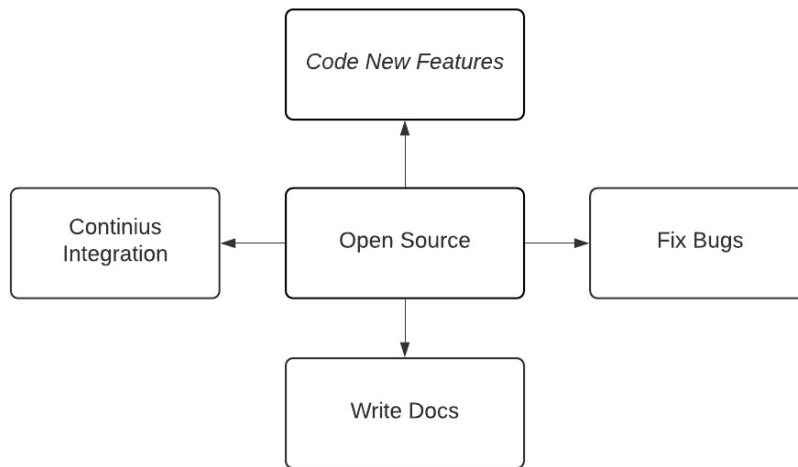


Figure 2: Open source software documentation configuration

Having a community guarantees bug-free and smooth-running applications and makes it possible for creators to come up with understandable and practical documentation. There are several benefits of preparing effective documentation:

- Communities are more tightly-knit as a result of cooperation
- Completes the application
- Makes it easy for the user to learn
- Contributes to the overall satisfaction of users
- Brings improvements to the user interface
- Increases the number of users who download the application

When an application lacks proper documentation, it ends up damaging the project's reputation and prompts users to look for alternatives.

While the existence of a community that consists of a large number of eager developers and enthusiasts makes it sound easy to prepare documentation, it is a challenge for any community. First of all, documentation must be written properly and understandably as not all users handle applications at a high level. Moreover, documentation must be reviewed and renewed as new updates for the corresponding applications are released. Although it sounds easy, most of the users complain about this issue. Some updates bring incremental improvements and performance boosts, but some change the whole user experience and add new features and improvements. For these updates, it is essential to renew the documentation in order to give the users a chance to use them effectively.

Software documentation can be done using several programming languages, including Python, R, C++, Java, and Ruby, and the documentation can be hosted on sites like GitHub (Turnbull, 2018).

Complete software documentation must include several key files, including:

- A ReadMe file, including concise information regarding the application, Instructions for installation, and brief guidance on how to use it
- Preparing, maintaining, and updating a dedicated wiki page
- Allowance of issue tracker for users.
- Prepare API documentation.
- Documentation of the written code
- Coding norms, such as file structure, comments, name conventions, programming methods, should be followed.
- Details for contributors
- Information about citation
- Information regarding licensing (Foster, 2020).

2.6. Open Source Movement

The open-source movement has been a force to reckon with in the internet world for the last two or three decades. Nowadays, Open source software communities play an increasingly influential role in the information technologies sphere. The model of such communities is to gather on the same internet platform and develop high-quality and competitive applications that can satisfy the needs of a wide spectre of users by providing them with the best possible user experience. The movement consists of individuals with shared values and norms, unique ethical approaches to the exchange of information between individuals (Carillo & Okoli, 2008).

Members of the OSS movement are encouraged to build communication with their fellows with the help of electronic communication. This includes participating in discussions on different platforms and forming communities through shared desire, which will then contribute to fostering strong relationships among participants. Eric Raymond famously described the OSS culture as a culture on the opposite side of typical Western self-motivated culture. As opposed to the Western culture, where people act based on their own interests, the open-source movement emphasizes the culture of giving away one's possession for free, and an individual's status in such communities is based on his or her willingness to give away things, rather than their control over them. As a result of the willingness to share knowledge and information without expecting anything in return, relationships between participants get better and more effective.

Another common characteristic between OSS communities is their proclaimed antipathy towards proprietary applications and big tech companies that develop and market them. Although the level of dislike varies, a common sentiment is that Microsoft and others are members of an "evil empire." There are also those who take a softer stance on this issue and believe in the possibility of "cohabitation" between proprietary and open-source software. However, overall, it is incorrect to call this attitude a rebellion. Most of the members of the movement want to write codes that run smoothly, are clean, and are out of corporate control.

There are several reasons why a developer may wish to take part in OSS projects. First and foremost, developers join such initiatives because of personal economic benefits. Their contributions may not go unnoticed and eventually lead to advancements in the job market, culminated by getting a well-paid

job in the industry. Additionally, members of OSS communities also get an ego boost by taking part in projects and making themselves known and appreciated among like-minded, skilled peers. At times, the need to join the community stems from the personal need of developers. This way, any problem that a said developer encounters in the application he or she uses will be fixed fast and effectively, with the help of other skilled participants of the community (Holmström, 1999).

Every successful (the keyword being successful) open-source project is backed by a community of developers and application users who keep it alive. While there are ways to make money from working on open source, one will find that many of these community members perform their duties to keep learning and share the knowledge out of love for the product and a sense of obligation to the community. Most people think that only professional developers or skilled engineers participate in OSS projects. While in reality, it is precisely the opposite. OSS communities thrive thanks to beginners because they can contribute to OSS. The contribution for OSS is not only to modify source code but all actions helping OSS developers. That is, indicating an error, annotating, and working documentation. 'Contributor' is a user helping OSS, and 'Committer' is a user having the authority to adjust source code. In some popular communities, a user can be a contributor after being active for more than two years. Many open-source applications reportedly have fewer bugs and problems thanks to the power of the community. While proprietary software is developed in tight circles and gets tested accordingly, open-source programs get tested extensively by nearly all community members, and the end result contains much fewer bugs than the proprietary applications (Mishra et al., 2002).

In the beginning, forming and being a part of a community was a hobby rather than a full-time job. However, nowadays, it is leaning towards being an excellent profitable business project. It is possible to popularize any given open-source application and capitalise on its fame by selling services like programming and education.

One of the offspring of Open Source movement is Community source. But being very similar to open source communities, community source differs itself from them by including specific organizations or educational institutions that contribute to the cause by employing experts or professionals who work in projects rather than committing voluntarily. This way, the human factor becomes more reassuring as projects are realized by people on a contract and certain legal obligations, so the project is managed more professionally. However, it is also true that the realized project will be strategically related to the needs of the institution or organization that it is developed by (Hanganu, 2014).

There is a duality in the nature of the Community Source Model. On the one hand, there is paid work which is against the voluntary work that is at the centre of the open-source philosophy. However, on the other hand, applications developed by such projects have all the essential characteristics of an open-source community project, such as the openness of the source code, the possibility to change, examine, redistribute the code or incorporate it into other products for free. Moreover, the end products of such endeavours are free to use by every individual and also organization.

The following is a general description of how Community Source Model projects work. Several institutions are aware that they are attempting to solve a common challenge, such as the requirement for a research administration system. After preliminary conversations and agreement on project objectives, deadlines, and philosophy, the establishments combine their assets under a project board of

institutional leaders. Institutions frequently agree to devote current staff time to the project's direction, so this is not a new monetary outlay but rather a virtual structure made up of existing staff. Each investor signs a Corporate Contributor Agreement, which grants the project or foundation a copyright license for the project (modelled on the practice of the Apache Foundation).

There are several noteworthy community source projects, such as Sakai and Quali. Sakai is educational platforms also known as Course Management System (CMS) or Learning Management system that encourages information exchange in teaching and research works. There are numerous academic establishments, commercial firms, and individuals who support the project, which was first released in 2005. More than a hundred universities and colleges use the platform. Its reliability and scalability help host more than 100,000 users from several different educational and academic backgrounds. In 2014, Sakai became an open-source project led by one company, distancing itself from its past community source roots (Hanganu, 2014).

Quali is another open-source venture that is based in Salt Lake City, USA. Company develops open-source administration software for universities and colleges in collaboration with higher education partner institutions. The project houses 59 institutions that actively participate in the development of Quali projects. Quali Student, Quali Research, and Quali Financial System are some of the most famous products developed by the project. 2012 saw Quali join the Apereo Foundation (Hanganu, 2014).

2.6.1. Open Source Architecture

Another result of the Open source movement philosophy, Open Source Architecture (OSArc) is a new reality in the world of Internet that creates new processes for designing, constructing, and operating buildings, infrastructure, and spaces. The foundations of this support were: avant-garde theory of architecture, science fiction, and language theory and open-source culture.

Open source architecture is a concept that combines the advancement of design and technology with the practices and ideas of open source projects to help rethink architecture design as a collective and collaborative effort. The focus of open source architecture is the potential interaction between the structure and its social, physical, and other environments, based on the people involved, rather than top-down solutions or linear processes. Transform the architecture to accommodate usage and an inclusive and transparent architecture (Quirk, 2013).

Today, a form of open-source language is the open architecture network created by Architecture for Humanity. It replaces the usual copyright restrictions with Creative Commons licensing and provides open access to drawings. Wider OSArc relies on digital assets and the interconnected spaces of the World Wide Web to deliver an instant collaboration that transcends established models of competition and profit. Familiar architectural tools such as blueprints and plans are being supplemented and increasingly replaced by interactive software applications that use relational data and parametric communication (Flynn, 2018).

OSArch does not just work on the production. It replaces static geometric shape architectures with the introduction of dynamic and collaborative processes, networks, and systems. Its supporters see the

differences between code and mass, relationships over compositions, networks over structures, adaptation over stasis. Its goal is to transform architecture from a top-down immutable delivery mechanism to a transparent, inclusive and bottom-up ecological system, even if it still includes top-down mechanisms (Botsman & Rogers, 2010).

OSArch is based on the work of not only amateurs but also experienced professionals. Like social software, it values each role of multiple users at all stages of the process: the clients, the community, and the designers. Democratic rules are at work here that reinforce the principles of open access and participation.

The open-source architecture revolutionizes every step of the traditional building process, from brief construction to demolition, programming and adaptive reuse, including the following:

Financing

New economic models make it possible to avoid the traditional client/architect/tenant model. As a result, private project financing now depends not only on one client but often at the expense of a more flexible framework than simple fees or taxation.

Betrothal

Traditional developments combine engagement programs in which the community is consulted on incoming developments. It does this through crude tools like focus groups. The use of space is often negotiated on terms set by the users.

Standards

They are very important to keep OSArc running smoothly and to make collaboration easier. Establishing common, open, modular standards (such as the mesh proposed by the OpenStructures project) avoids the problem of hardware compatibility and interface between components.

Design

Parametric design tools such as Grasshopper, Generative Components, Revit®, and Digital Project enable users to interact and modify, test and navigate virtual designs, than just consumers. Open source codes and scripts enable development communities to share and compare information and collectively optimize production with modular components, accelerating the accumulation of common knowledge over time.

Construction

Operating systems are built to manage the design, construction, and deployment phases. Materials will report their condition during manufacture and construction. This will ensure that the database is validated and committed, and distributed throughout its life.

Employment

OSArc allows residents to control and shape their personal environment. Fully intelligent network spaces constantly transmit their various properties, states, and attributes - often through decentralized systems. Personalization replaces standardization because rooms "intelligently" recognize and respond

to individual occupants. Real-time monitoring, feedback, and display of the environment are becoming integral elements of the continuous life of spaces and objects. Maintenance and operation are becoming an integral part of the construction process; in a world of growth and change at OSArc, a building is never 'complete' (Leadbeater, 2008).

2.7. Difference between OpenBIM and OSS

Most people are confusing these two concepts as they are so similar to each other. Open-source software does not equivalent to the OpenBIM definition. Both of them correspond to the needs of BIM but in a different context. Open-source software is just a tool or application that helps the user reach the BIM requirements, while OpenBIM is a collaborative process neutral to the used software or application. The OpenBIM process can be defined as shareable project information, enabling all project participants to collaborate seamlessly. OpenBIM promotes interoperability and benefits projects and assets throughout their lifecycle.

In recent years, specialized software for BIM has become widespread in the architecture, engineering, and construction industry (AEC). It processes the received information on the model and visualizes the project.

Building information modelling (BIM) software provides users with communication and design. With the help of BIM, it is easy to understand how the building will look, its characteristics, and its cost. Furthermore, all this is possible to gain in the process of design. BIM technology took an essential role in the 3D reconstruction of cultural heritage monuments. Software supporting BIM provides users with tools, concept analysis, design data support with documentation, and 3D modelling. As a result, there is a demand for the BIM software industry development to be able to fully support construction, reconstruction, restoration, and management of 3D models of cultural heritage objects. To fulfil this task, not only commercial software is used, but also free and open-source software is part of this market.

In recent years, the popularity of open-source has grown. However, commercial software is often preferred because the OSS is not complete and does not cover all stages of the BIM process. Sometimes, to finish one project, the team is forced to use more than one software or plug-ins or write a script to do the task. For all its advantages, FOSS also has its limitations. However, in recent years, FOSS for BIM has evolved and expanded. What are these restrictions:

1. Sometimes data inconsistency is observed.
2. Requires programming skills and experience to customize the app.
3. There may also be less support and there is also sometimes no user manual.
4. Although OSS is free, it can still be associated with some indirect costs, like additional functions.
5. Small business skills may not be sufficient for some of the BIM OS software and tools.

OpenBIM

OpenBIM increased the benefits of BIM; it boosted the availability, usability, governance, and resilience of digital data in the built asset industry. In addition, OpenBIM processes combine the cooperation of all project participants and all information about it; it simplifies communication and thus benefits projects and assets.

OpenBIM provides the ability to develop new ways of working by transforming traditional peer-to-peer workflows. OpenBIM improves asset performance by disrupting the data warehouse. Companies working with OpenBIM are creating cross-party collaboration methodologies and improving communication and exchange standards (Baldwin, 2017). This way, they get better project results, more predictability, improved performance, and increased security with less risk.

Throughout the project, OpenBIM connects people, processes, and data to deliver, operate, and maintain assets. With OpenBIM plus seamless digital workflows, critical project information is made available to employees to support decision-making. It is possible at all project stages from inception to handover, refurbishment, and even demolition. OpenBIM eliminates BIM data problems that are usually limited to proprietary formats, discipline, or project phases.

Without departing from international standards and workflows, OpenBIM increases the breadth and depth of BIM use. Technical applications developed for OpenBIM enhance data management and eliminate disconnected workflows. OpenBIM allows digital workflows based on vendor-agnostic formats like IFC, BCF, COBie, CityGML, gbXML, etc.

OpenBIM also provides an affordable digital twin that provides the foundation for a long-term data strategy for the assets built. This allows projects to remain sustainable and to manage the built environment effectively.

The OpenBIM Principles recognize that:

- Interoperability is central to digital transformation.
- Open and neutral standards need to be created to facilitate communication.
- The reliable exchange of information depends on various unrelated criteria.
- Collaboration workflows are progressing through open and flexible data formats.
- Technology flexibility provides excellent value to all stakeholders.
- Long-term data interoperability standards create sustainability.

The benefits for the constructed asset industry are as follows:

- OpenBIM truly improve the project collaboration for project delivery
- OpenBIM provides improved asset management.
- OpenBIM provides access to BIM data, which is generated during design, throughout the entire project life cycle
- OpenBIM expands the breadth and depth of BIM results by creating a consistent consensus and language while adhering to international standards and generally accepted workflows.
- OpenBIM fosters a common data environment that empowers users to develop new workflows, software applications, and technology automation.

- OpenBIM provides an affordable digital twin that provides the foundation for a long-term data strategy for the assets built (Petrie, 2017).

2.8. Advantage and Disadvantage of OSS

The benefits of open-source software (Regoli, 2015):

1. Price

Practice shows that business owners save about \$ 60 billion annually with open-source software. Such information seems implausible, but these programs are designed in such a way that anyone, especially those who cannot afford to buy commercial products, can use them. First, because they are often free, even with additional downloads. And secondly, many programs are designed in such a way that any computer supports them, which means it will extend the life of your old equipment. You don't have to replace it from time to time.

2. Talented and intelligent developers.

It is a mistake to think that large, established software companies hire the best talent. We think so because they have the financial ability to select the best developer in the industry to create the best product. So, we usually buy a product at a higher price, believing that its quality will be better.

Really large companies work with experienced and highly qualified people. But at the same time, such software specialists do not always care about money. Many consider themselves valuable not only because of their salaries, but because they strive to create a program that will generate praise and approval, and also change the world for the better. In this case, more open-source software develops, because there everyone embodies their ideas and creativity. There are no strict restrictions in the rules, as in the corporate world. Therefore, specialists are free to experiment. Thus, the consumer has access to modern world-class software. And he pays practically nothing.

3. Reliability.

Open-source software is often developed by experienced and talented people. They, in turn, try to do everything possible to ensure that the programs are of high quality. Since not one person or even two is involved in the development, there are many eyes that will cope with mistakes and many pairs of hands that correct these shortcomings in the shortest possible time. Thus, the product becomes of excellent quality, has many useful functions, and works perfectly.

4. Flexibility.

When you are not limited to a proprietary product, you are not adhering to a specific IT architecture. It, in turn, often requires software and even hardware updates. To avoid this, you can mix and match your software. This will create an overall IT infrastructure that meets all of your needs. With a wide range of features on the market, you just look and download those that meet your needs and specifications.

Disadvantages of open source software

1. Available to intruders.

Due to the fact that everyone has access to the source code of open-source software, the problem arises that not all users have good intentions. Indeed, there are people who strive to identify the defect and make improvements to the program. But many, on the contrary, use access to create errors. And there is also the possibility of equipment infection, identity theft. This rarely happens because the companies that are in charge of the software strictly control the quality and guarantee the quality when they are released to the market.

2. Less convenience than commercial versions.

This is not the case for all open-source software. For example, LibreOffice, Firefox Mozilla, and the Android operating system are, on the contrary, simple. But there are other programs that are created specifically for the specific wishes of the developer. Therefore, often little attention is paid to the user interface of the software and therefore makes it difficult to use. Especially, it is difficult for those who are not versed in technology.

3. No support.

More often than not, users prefer commercially released programs because they give them peace of mind. After all, they know exactly who and how they developed and distributed the product. There is a specific person responsible for the same operation of the program and is responsible for all equipment damage. There is no such support for open-source software because it is developed by several people at the same time.

But, at the same time, even large software companies disclaim responsibility. When you read the end-user license agreements, you will notice that the developers are disclaiming obligations, and the responsibility for the product lies with the buyer. That is, you will just spend money, but you will not get support.

Thus, you have to compromise on responsibility and payment (FOSSA, 2021).

3. ANALYSIS OF OPEN-SOURCE SOFTWARE FOR ARCHITECTURAL DESIGN

3.1. OSS in Architectural Design

The increased number of open-source software products aimed at different fields, including architecture and construction, shows significant demand. This trend motivates further inquiry into the progress of isolated efforts in that area. Nevertheless, the number of proprietary software tools in existence is significantly greater than those of their open-source alternatives.

Open-source projects also have the advantage of bridging the gap between the end-users and the developers of the product because of the transparent nature of the development process. This openness allows the end-users to contribute directly towards the engineering of their tools. As a result, the final product is shaped after their real needs, and the tools complement their techniques. The involvement of architects in open-source projects has yielded noteworthy products in the past.

Unfortunately, designers and architects in many firms do not use open-source software. And there are several explanations for this. One is the use of open-source requires more technical competence, usage of add-ons, and knowledge of scripting, therefore, it is easier to pay for a functional package and use it. Today, OSS provides so many features to support the architectural project, making it compete with paid software, especially for design. However, some Architects and designers disagree that open-source software can deliver the same level of performance as paid software.

Advanced CAD (Computer-Aided Design) and BIM (Building Information Modelling) software help translate ideas into detailed designs. They show detailed and realistic results. They are good at timing and flexible enough to modify the model accordingly. Architectural design software has complex interfaces. They are integrated programs used by architects to solve complex structural problems such as gravity levels and check for any weak points in the structure. Architectural design Applications differ by the function or the stage of design where they are used. According to the RIBA plan of work (RIBA, 2020) each step of the project is organized to stages to provide framework for design and construction process. There are 8 stages for plan of work. Every stage can use different software due the work to be done. If to take to consideration these stages for design it is possible to organize the software into 2 categorizes according to their functions:

Strategic stage / Preparation stage / Concept stage

- Concept Design
- Presentation
- GIS software

Spatial Development / Technical Design / Construction Documentation

- 2D CAD
- 3D CAD / BIM
- Analysis (structural, Sustainable, Energy, etc.)

According to the categories shown above, it is possible to introduce a list of FOSS software used in architectural design stages. The table is organized by five definitions for listed applications. These definitions are (1) functional purpose, (2) the computer language used to develop them, (3) the license that they are registered, (4) last update date (no software with old realize date is included in the list, as they are considered as abandoned projects), and (5) documentation information, which shows how to use the software and how its development history.

Table 1. Open-source software in the Architecture and Design field

Software name	Functional purpose	Computer language	License	Last update date	Documentation
Archipak	Computational Modeller	Python	GPL -3	2021/03/20	Webpage
BIMserver	Model data management	Java	AGPL-3.0	2020/02/04	Manual, Forum
LibreCAD	2D CAD modeller	C++	GPLv	2020/12/31	Wiki, Manual, Forum
FreeCAD	CAD / BIM modeller	C++, Python	LGPL	2021/04/22	Wiki, manual
Sweet Home 3D	Schematic design modeller	Java	GNU, GPL	2021/07/27	Webpage Forum
Dynamo BIM	Computational Modeller	C#	Apache	2019/08/19	Wiki, Manual
IfcOpenShell	Model data analysis	C++	LGPL	2021/09/03	Forum, Webpage
BlenderBIM add-on	CAD / BIM modeller	C, C++ Python	LGPL-2	2021/06/05	Wiki, Forum
BRL-CAD	CAD modeller	C,C++,TCL	LGPL-2, BSD	2021/02/6	Wiki, Forum
BHoM	Computational Modeller	C#, PowerShell,HT ML	LGPL-3	2021/06/24	Webpage, Forum
Blender	Visualization Modeller	C, C++, Python	GPL	2021/09/01	Wiki, Manual
OpenSCAD	CAD modeller	C++,Qt, CGAL, OpenCSG, OpenGL	GPL-2.0	2021/08/21	Wiki, Manual
COMPAS	Computational Modeller	Python	MIT	2021/01/18	Webpage
TAD	CAD / BIM modeller	Visual Prolog, Delphi	TAD License Version 1.2	2021/08/17	Webpage
BlenderGIS	GIS modeller	Python	GPL-3	2020/11/25	Forum

Excalidraw	Schematic design modeller	TypeScript	MIT	2021/05/03	Forum
Krita	Presentation modeller	Python	GPL-3	2021/08/25	Webpage
Gimp	Presentation modeller	C, GTK+	GPL-3	2021/03/29	Webpage, Manual
Homemaker add-on	Schematic design modeller	Python	GPL-3	2021/08/07	Forum
DepthmapX	Analysis Software	C++	GPL-3	2020/11/09	Forum
Housing_Deficit	Analysis Software	R	CC-IGO 3.0	2020/08/25	Forum
urbanpy	Analysis Software	Jupyter Notebook	CC-IGO 3.0 BY-NC-ND	2021/02/25	Forum
Appleseed	Presentation modeller	C++	MIT	2019/09/3	Webpage, Forum
Blender4web	Presentation modeller	C, Python, JavaScript, C++	GPL-3	2021/05/18	Webpage, Manual
Inkscape	Presentation modeller	C++	GPL-2	2021/05/24	Webpage, Forum
LuxRender	Presentation modeller	C++, Python	Apache-2	2021/04/11	Wiki, Web Page, Manual
Mitsuba	Presentation modeller	C++, Python	GPL-3	2020/07/27	Webpage, Forum
QCAD	2D CAD modeller	C++	GPL-3	2021/07/20	Webpage, Forum, Manual
Solvespace	2D/3D CAD modeller	C++	GPL-3	2021/04/18	Webpage, Forum
Sverchok	Computational Modeller	Python	GPL-3	2021/06/30	Webpage, Forum
Topologic	Analysis Software	C++, C#	AGPL-3	2019/02/01	Webpage
BIMsurfer	Model data analysis	JavaScript	MIT	2020/02/04	Forum
Ladybug	Analysis Software	Python	AGPL-3	2021/09/03	Webpage, Forum
GanttProject	Management modeller	Java, Kotlin	GPL-3	2021/05/18	Webpage
OpenProject	Management modeller	Ruby, TypeScript	GPL-3	2021/07/29	Webpage, Forum, Manual
IFC.js	Management	JavaScript	MIT	2021/09/06	Forum

3.2. BIM Software Expectation

BIM is a process characterized by the use of specialized software tools to realize the effective design, documentation, and implementation of the intentions of all stakeholders in the AEC project. This article mainly discusses the software aspects of BIM tools and their use, mainly in the design phase of construction projects. The secondary consideration is the interoperability of their participation in the collaboration phase.

In the discussion of standard definitions, software vendors agreed that: Architectural BIM applications should be able to create virtual three-dimensional (3D) objects the same as those in the AEC industry. The virtual objects created must have industry standard attributes, and designers can modify these attributes to create their own variants. Therefore, they should be parameterized.

A change to a virtual object should notify the corresponding change or update of the object in its hierarchy. The objects must have intelligent relationships. The child object must be converted together with the parent or host object. For example, holes in wall objects need to be placed along with the doors that need them or disappear when the relationship ceases to exist (perhaps if the doors are removed or removed from the wall). Similarly, changing the height property of a door should trigger an understandable re-adjustment of its sub-objects. That is, panels, frames, and glass should be updated accordingly. Architectural BIM applications must provide methods for documentation, conceptualization, and visualization. BIM applications should be able to convert their native file structure into a standard format to facilitate interoperability with other software tools that may participate in the BIM process. This translation should be done without significant losses to promote collaboration between professionals in the process.

3.3. Detailed analysis of Open-Source applications

Four design programs from the table will be analyzed in detail according to architectural design and BIM expectations to examine if these applications can help create professional-level projects. These four programs are FreeCAD, BlenderBIM add-on, LibreCAD, and TAD Designer. ALL four programs have different development directions. FreeCAD initially was parametric software for mechanical design, while BlenderBIM add-on is offspring of graphical visualization software Blender. LibreCAD is a 2D CAD program, which cannot be considered a BIM application, but it can support other software with construction documentation and precise drawings. However, TAD is a different story. In contrast with the other three software, TAD was created by an architect for architecture. The idea of TAD is to support architects in the conceptual phase of the project. In other words, it helps architects with the early stage of the project. TAD is not open-source software, but still, code can be distributed under registration and agreement with the TAD developer. In 2020, the developer of TAD announced that the next release of the software would be with open source code.

3.3.1. FreeCAD

FreeCAD - is an open-source and totally free, parametric 3D modelling application which allows designing elements and object with precision. FreeCAD can run on Windows, Linux, and Mac OSX systems keeping all the functionalities equally same and having the same interface look for all three systems; this makes FreeCAD a multi-platform application. FreeCAD uses different types of open

source libraries such as Open Cascade Technology (OCCT), QT, Coin3D, and the primary scripting language Python. Also, FreeCAD by itself can become a library for new applications and programs. FreeCAD by structure is feature-based parametric modular software that lets users add or create new functionalities without making changes to the core system. Even though FreeCAD was initially created as a mechanical engineering or product design software, it still can be used in architecture, 3d printing, engineering, and other fields (FreeCAD, 2021).

FreeCAD has been in development since 2001 and boasts a long list of capabilities. Although several features are missing, it is powerful enough for hobbyists and small workshops. In addition, the FreeCAD community has a rapidly growing network of enthusiastic users; like all free software projects, FreeCAD relies on its community to grow, add features, and fix errors.

History - The history of FreeCAD began in January 2001, when Jürgen Riegel began working on a project that would later officially become FreeCAD. Jürgen Riegel started to work on GOM (Graphical Object Modular) with the core use of Python, Qt and Cas.CADE, which was a commercial software and also included geometric modelling engine (CAD engine) and short time after in 2000 was released with an open source license. In 2002 17th of March Jürgen registered FreeCAD as software. April 2003, Werner Meyer, another key developer of FreeCAD joining the company Imetric (FreeCAD, 2021). And these new contacts give a huge opportunity to develop FreeCAD since they was searching for their 3D sensors a new 3D software platform. 2005 saw Imetric give most of its Mesh Module to FreeCAD and the Open Source community, and FreeCAD became the base of the sensor system software they used. In 2005, the developers replacing OpenCascade document framework with their own implementation and keeping only the CAD kernel of OpenCascade. In 2007, developers switched to QT4 and, therefore, to the LGPL. In 2008, Yorik van Havre joined the project as 3rd leading developer and worked mainly on Draft Module, expanding the FreeCAD documentation and creating many icons for the FreeCAD GUI and defining its style. The first version to have Draft Module was version 0.7 which was released in 2009 April. Before, to draft 2D geometry was impossible as it was hard to create it by using GUI, so solution came with the use of Python instead of C++ which is the main programming language for FreeCAD. The Part module provided a basic CSG process, allowing users to create primitive forms and perform Boolean operations using the Part menu. It was also feasible to extrude 2D profiles and fillet them. The new Draft workspace has proven that Python integration has been successful and can be used to extend or customize FreeCAD's capabilities. By the end of 2009 FreeCAD was already a Debian package in the Debian archives. In 2010, FreeCAD was included to the Ubuntu 10.04 libraries. The FreeCAD Maintainers team was formed in early 2011, using the Launchpad web platform to provide new bug-free versions of FreeCAD as well as daily build packages to Ubuntu users. In January 2012, version 0.12 was released, which included a more comprehensive Sketcher workbench. FreeGCS, a completely redesigned solver, was supplied.

License – Two licenses are used for FreeCAD. LGPL2+, the second version of Lesser General Public License is applied to the application itself, while Creative Commons Attribution 3.0 is in use for the documentation purposes. FreeCAD is a totally free and non-restrictive application which gives the total freedom to the user. Once the user downloads the application, that copy belongs to him/her and there is no contractual or legal obligation. Unlike many other applications, there is no obligation to

update FreeCAD if the user doesn't wish to. Because the FreeCAD source code is open to the public and can be examined, it is easy to verify that it does not perform actions without the consent of the user, such as sending confidential data to a third party (FreeCAD, 2021).

FreeCAD can be used free of charge for any purpose, whether it's personal, corporate, or institutional. Furthermore, the user is free to customize and adapt FreeCAD to his or her own needs. It's also feasible to utilize FreeCAD as a foundation for constructing new applications, or simply to extend it by adding new modules. If a user creates a module to be used as an extension and does not include any FreeCAD code, they can choose any license they choose. If FreeCAD is used to create new application, User can choose either the GPL or the LGPL license, or any other license that is compatible with LGPL, to allow the use of his/her work in proprietary software or not. Also, User not forced to make his/her application open source. The LGPL license, however, ask for two basic things:

- 1) To inform users that application is using FreeCAD and that FreeCAD is LGPL-licensed
- 2) To separate your new application from the FreeCAD components.

That is usually done by either dynamically linking to the FreeCAD components, so users are allowed to change it, or making the FreeCAD source code, along with the modifications brought to it, available to new users. If the user utilizes FreeCAD in order to develop a new program, they can select between the GPL and the LGPL licenses, or any other LGPL compatible license, to allow or disallow the use of his or her work in proprietary software. Furthermore, it is not obligatory for the user to make their source code available. The LGPL license, on the other hand, stipulates two requirements:

- To warn users that the program uses FreeCAD and that FreeCAD is licensed under the LGPL
- To segregate your new application from the FreeCAD components.

Supported file formats - The FreeCAD basic file format is FreeCAD's primary standard file format (.FCStd). It is a composite format that allows for the compression and embedding of many types of data. FCStd is a regular zip file that contains one or more files organized in a certain way. For exploring it like a normal directory, it is enough to change the extension to .zip. The Document.xml and GuiDocument.xml files, as well as any number of .brp (BREP) files are at the root of the package. The thumbnail may be stored in one subdirectory, and the SVG templates utilized by TechDraw in another.

```

File.FCStd (File.zip)
|
|--thumbnails/
|
|   |--Thumbnail.png
|
|--Templates/
|
|   |--MyPage.svg
|
|--Document.xml
|--GuiDocument.xml
|--Shape1.brp
|--Shape2.brp
|--etc.

```

Figure 3: Structure of a typical .FCStd file (Anon., 2021)

All geometric and parametric object definitions are contained in the Document.xml file. The visual representation details of items are then stored in GuiDocument.xml. Object brep-files and thumbnails of drawings are among the other files.

Aside from FreeCAD's own file format, the following file types can be exported and imported:

Table 2: FreeCAD import and export file formats (FreeCAD, 2021).

Format	Description	Import/Open	Export/ Create
FCStd	FreeCAD native format	Std Open, StdMergeProjects	Std Save
STEP	Exchange format for engineering models	Std Import, Part Import	Std Export, Part Export
.iges	Older solid-based format	Std Import, Part Import	Std Export, Part Export
.brep	OpenCasCade native format	Std Import, Part Import	Std Export, Part Export
.csg	OpenCasCade native format	Std Import	Std Export
.gcode	G-code	Std Import	PathPost
.obj	Wavefront format	Std Import	Std Export
.stl	Stereolithography mesh (mostly used for 3D printing)	Std Import, Mesh Import	Std Export, Mesh Export
.dwg	AutoCAD® native format. Only 2D geometry is supported. External software required	Std Import	Std Export
.dae	Collada format. For Linux users: External pyCollada module required.	Std Import	Std Export
.dxf	Autodesk drawing exchange format. Only 2D geometry is supported. External software required for the legacy Python importer and the legacy Python exporter.	Std Import	Std Export, TechDrawExportPageDXF
.ifc	Industry Foundation Classes	Std Import	Std Export

	exchange format for BIM models. For Linux users: External IfcOpenShell module required.		
.svg	Scalable vector graphics format	Std Import	Std Export, TechDrawExportPageSVG
.vrm	VRML Web 3D format	Std Import	Std Export, Mesh Export
.ply	Point Cloud format	Std Import, Points Import	Std Export, Points Export
.pov	Povray format	Std Import	Raytracing WriteCamera, Raytracing WritePart, Raytracing WriteView

Community – The FreeCAD community is one of the strongest and biggest among the open-source communities. It is possible to claim that FreeCAD is developed and maintained by one community. The community recruits everyone with any level of knowledge, starting from beginners, and everyone can take part in the development of the software.

The FreeCAD community is growing at a very fast pace. Different groups of people from all over the World are united there, and all work for FreeCAD voluntarily. Among them, some are professional programmers, some are long time FreeCAD users, and eventually, they end up knowing a lot about FreeCAD programming and many are new members who start to explore FreeCAD.

FreeCAD forum is the principal place for members to come together to meet and discuss topics. Moreover, the forum is an excellent opportunity for new FreeCAD users to ask questions if they need help. Usually, new users get several replies within the same hour, which shows that the community works very operatively. Also, here users can share their achievements with FreeCAD to help new members and share their opinions on their experience with the technical development of software. All the FreeCAD development is discussed on the forum, and anybody is free to read or participate. Also, FreeCAD forms other communities outside of the forum, in different social media platforms like Facebook, Youtube, and Github, which helps spread knowledge about this software (Manual, 2021).

The major duties listed on the FreeCAD community form are for everyone, programmers and other community members, and include things like the major duties listed on the FreeCAD community form are for everyone, programmers, other community members, and include things like:

- Assist newcomers with their queries and educational needs
- Reporting defects, as it is hard for developers to test and repair all possible use cases; Members of the FreeCAD community contribute to the wiki page. However, several portions are still missing or require changes or updates.
- Translation of the FreeCAD documentation and wiki page into other languages. Thanks to the efforts of the community, the wiki page of the application now supports languages other than English.

- Scripting and macro creation. FreeCAD requires assistance with functionality and the development of workbenches and add-ons as the number of Macros grows.
- Programming in Python and C++

Documentation

FreeCAD's documentation system is well-organized and well-developed. To document the project, the developers employed three main tools:

- A dedicated FreeCAD wiki page,
- Video tutorials,
- Online manual.

Interface - The FreeCAD user interface is built using the contemporary Qt toolkit and has cutting-edge structure. Because this framework is built to be utilized in a wide range of applications, the FreeCAD interface is extremely traditional and easy to figure out.

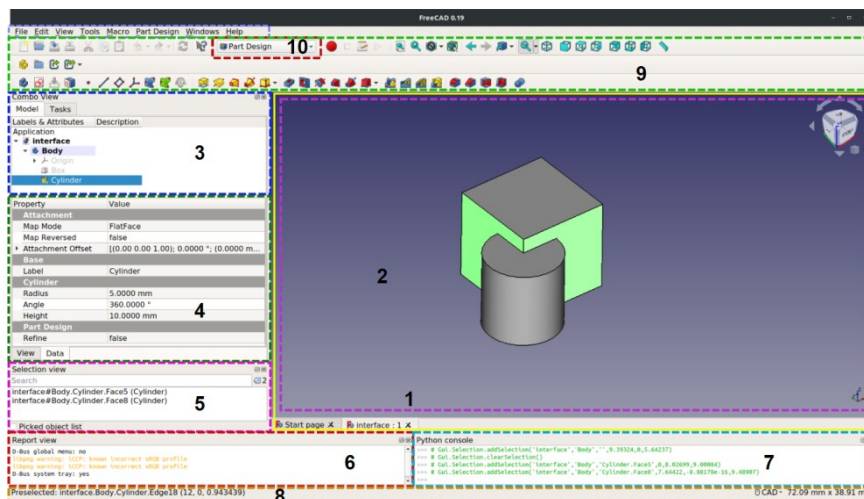


Figure 4: Standard FreeCAD interface in 0.1 (FreeCAD, 2021)

1. The main area of view, which contains tabbed windows, the most important of which is the 3D view.
2. The 3D view, which displays the document's geometrical objects.
3. The tree view (part of the combo view), which displays the document's structure and development history, as well as the task panel for active commands.
4. The property editor (part of the combo view), which allows you to see and change the properties of the items you've selected.
5. The selection view, which displays the objects (vertices, edges, and faces) that have been selected.
6. Messages, warnings, and errors are displayed in the report view (or output window).
7. The Python console, which displays all of the commands that have been executed and allows you to type Python code.
8. The status bar, which displays some messages and tooltips.
9. The toolbar area, which houses the docked toolbars.
10. The workbench selector, which allows you to choose the active workbench.

11. The normal menu, which contains the program's main functions.

Customisation of the interface

FreeCAD has a very adjustable user interface. All panels and toolbars can be moved about or layered on top of one another, as well as closed and reopened if needed. However, there are numerous other alternatives, such as constructing bespoke toolbars with tools from any of the Workbenches or assigning and altering keyboard shortcuts. The menus and toolbars that come with FreeCAD and its workbenches, on the other hand, cannot be modified (FreeCAD, 2021).

These advanced customization options are available from the **Tools** → **Customize menu**:

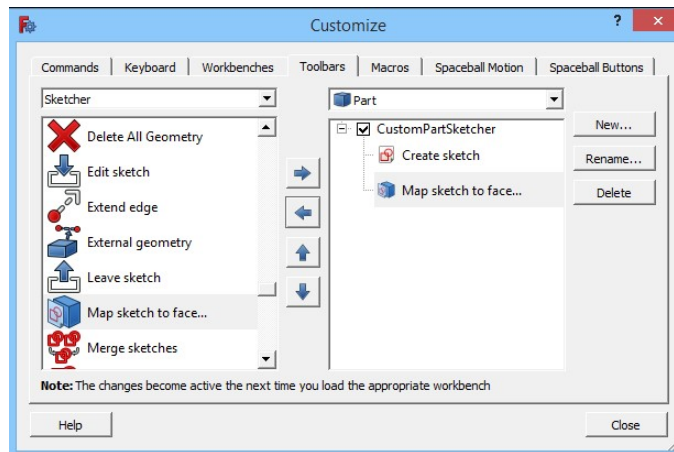


Figure 5: Customize menu (FreeCAD, 2021)














Workbenches










FreeCAD is based on the Workbench concept and is extremely comparable to other design programs like Revit® or CATIA. Workbenches are collections of tools (toolbar buttons, menus, and other interface elements) arranged together for a certain purpose, such as mesh editing, 2D object drawing, or restricted sketching. While working on the same project, you can go from one Workbench to another. When you switch from one Workbench to another, the tools available on the interface change. The contents of the scene do not change, but the toolbars, command bars, and perhaps other aspects of the interface switch to the new workbench. For example, start creating 2D shapes with the Draft Workbench, then continue working on them with the Part Workbench (FreeCAD, 2021).

Workbenches are also entirely configurable, with the ability to integrate tools from other workbenches or even self-created tools (known in FreeCAD as Macros). A single tool may exist on many workbenches. Regardless of the workbench a tool appears in, its button icon will always be the same.

It's worth noting that a Workbench is sometimes referred to as a Module. Workbenches and Modules, on the other hand, are two distinct entities. A Workbench is a specific form of Module with a GUI setup, whereas a Module is any extension of FreeCAD (toolbars and menus). A table of existing FreeCAD workbenches and their descriptions is shown below.

Table 3: FreeCAD workbench list (FreeCAD, 2021)

Icon	Name	Description	Status
	Arch Workbench	Working with architectural elements.	Constant Development
	Draft Workbench	Contains 2D tools and basic 2D and 3D CAD operations. PDevelop provides tools to define a working plane, a grid, and a snapping system to precisely control the position of geometry.	Developed
	Drawing Workbench	The Drawing module allows to put 3D work views of models in a 2D window and insert that in a sheet to print	Deprecated
	FEM Workbench	Provides Finite Element Analysis (FEA) workflow. All tools to make an analysis are combined into one graphical user interface (GUI).	Constant Development
	Image Workbench	Working with bitmap images. A major use of this workbench is tracing over the image, with the Draft or Sketcher tools, in order to generate a solid body based on the contours of the image.	Developed
	Inspection Workbench	Specific tools for examination and comparing of shapes.	Under Development
	Mesh Workbench	Working with triangle meshes	Developed
	OpenSCAD Workbench	Offer interoperability with OpenSCAD and repairing constructive solid geometry (CSG) model history.	Limited Development
	Part Workbench	Working with CAD parts. Based on a constructive solid geometry (CSG) methodology for building shapes.	Under Development
	Part Design Workbench	Modelling complex solid parts from sketches. Uses a parametric, feature editing methodology, a basic solid is sequentially transformed by adding features.	Developed
	Path Workbench	Produce machine instructions for CNC machines from a FreeCAD 3D model. Generate G-Code instructions.	Under Development
	Points Workbench	Working with point clouds	Under Development
	Raytracing Workbench	Generate renders by processing with an external renderer. Two renderers are supported: POV-Ray and LuxRender. Made in C++	Deprecated






	Render Workbench	Replacement for the built-in Raytracing Workbench. Written fully in Python. Provides enhanced features, compared to Raytracing. Supports - Pov-Ray, Intel Ospray Studio, Appleseed, LuxCoreRender, Pbr v4 and Cycles.	UnderDevelopment
	Reverse Engineering Workbench	Provide specific tools to convert shapes/solids/meshes into parametric FreeCAD-compatible features.	UnderDevelopment
	Robot Workbench	Studying robot movements. Is a tool to simulate a standard 6-axis industrial robot	Unmaintained
	Sketcher Workbench	Working with geometry-constrained sketches, allowing 2D shapes to follow precise geometrical definitions in terms of length, angles, and relationships.	Developed
	Spreadsheet Workbench	Creating and editing spreadsheet data and export the data to other spreadsheet applications such as LibreOffice or Microsoft Excel.	Developed
	StartCenter Workbench	Is not really a workbench, it's just the page that is presented when you open FreeCAD with no document loaded.	Developed
	Surface Workbench	Provides tools to create and modify NURBS surfaces. It is similar as the Part Shape builder tool's Face from edges option. . But tools in this workbench are parametric and provide additional options.	Developed
	TechDraw Workbench	Producing technical drawings from 3D models. It is the successor of the Drawing Workbench. The resulting drawings can be used for things like documentation, manufacturing instructions, contracts, permits, etc.	UnderDevelopment
	Test Framework Workbench	Is not really a modelling workbench, but it contains a set of Python scripts to perform different tests on the core components of FreeCAD, in order to debug problems.	ConstantDevelopment
	Web Workbench	Provides with a browser window instead of the 3D view within FreeCAD.	Developed








External Workbenches

Workbenches produced by expert users that have not yet been merged into the core FreeCAD source code are known as external workbenches. These workbenches have not been tested and can be used with all versions of FreeCAD because the FreeCAD core development team does not support them.




Questions, bug reports, and enhancement requests should be directed to the workbench's author. Python makes it simple to program the FreeCAD workbench, and most of them can be installed directly from FreeCAD using the plug-in management. The plug-in management is where all external workbenches and macros are installed. In general, an add-on is any component that is not included in the FreeCAD base installation but that the user can add. Plug-ins is divided into three categories: First, there are Python code macros and small snippets that provide new tools or functions in a single file. The second is a single Python source file module, or a set of Python files, that adds functionality to the product. The module may or may not offer a graphical "workbench," but it may provide accompanying functions such as a library for performing format conversions or modifying graphical interface code. The Workbench set of Python files is the most recent addition. These files provide important graphical user interface commands (tools) for various themes, such as tools for creating cabinets, tools for construction, or tools for designing ships, and so on. These workbenches usually create new toolbars with buttons for commands. The first is FreeCADdons and FreeCADmacros, and the second is macro recipes, which are both storage libraries for macros and external workbenches. AEC Field is represented in the table below by a list of external workbenches.

Table 4: List of external workbenches for FreeCAD (FreeCAD, 2021)

Icon	Name	Topic	Discription	Status
	ArchTextures	Architecture and construction	It allows you to add basic, non-photorealistic textures to architectural objects created with the Arch Workbench.	Maintained
	BCFPlugin	Architecture and construction	It aims to support the BIM Collaboration Format (BCF).	Maintained
	BIM	Architecture and construction	It aims to implement complete building information modelling(BIM) workflow in FreeCAD. It extends the Arch Workbench, and gathers tools from other workbenches to provide an environment that is convenient to model buildings, and work with IFC files.	Maintained
	BIMBots	Architecture and construction	It allows you to upload a FreeCAD model or selected parts of a FreeCAD model to a BIMBots instance (usually a BIMServer with external services enabled), and perform different services and analyses on your model, and read the results in FreeCAD, usually under the form of a text report, or a BCF file.	Maintained
	Dodo	Architecture and construction	It provides tools to create frames (trusses, beams) and pipelines (tubes, elbows, flanges), and query those objects. This is the new version of Flamingo, intended for Python 3 and Qt5.	Maintained

	GeoData	Architecture and construction	It provides tool to import geographical information from a given point on Earth by its latitude and longitude, of from OpenStreetMap, Google Maps, Bing Map, or Here Map.	Unmaintained
	Geomatics	Architecture and construction	It is partially based on the GeoData. It provides functionality specific to Geomatics or Survey engineering, including importing point files, creating surfaces, creating contours, and creating sections. This is partially migrated to the Trails workbench.	Unmaintained
	OSE Piping	Architecture and construction	Create different piping fittings. It supports Flamingo.	Maintained
	Reinforcement	Architecture and construction	It provides tools for Reinforcement Generation and Detailing. This workbench provides an interface and presets for the creation of common rebar types. And tools to generate rebars bill of material, rebar shape cut list, bar bending schedule, and rebars drawing and dimension.	Maintained
	SteelColumn	Architecture and construction	It provides tools for creating complex steel columns assembled from IPE profiles and plates.	Maintained
	Timber	Architecture and construction	It provides tools to facilitate the design and modelling of wood-frame and structural walls. This workbench is no longer developed nor maintained by its author.	Unmaintained
	WoodFrame	Architecture and construction	It provides tools to facilitate the design and modelling of wood-frame and structural walls, as well as cut lists for beams.	Unmaintained
	Trails	Architecture and construction	It provides functionality specific to transportation engineering (roads and rail). It includes components to perform analysis of curvature.	Maintained
	CADEXchanger	Information and data	It is an extension that allows FreeCAD to import and export file formats supported by the commercial "CAD Exchanger" application, such as Rhino 3dm or ACIS sat, and mesh formats like OBJ and STL.	Maintained

	dxf_library	Information and data	<p>Install the Python DXF importer and exporter. This is required in FreeCAD version v0.15 and earlier. This operation is no longer required when using the built-in DXF importer in version 0.16 and higher.</p> <p>If you want to explicitly use the Python importer or export directly from a 3D model, you still need this library. Note that the built-in importer is faster than the Python importer, but in many cases, the Python importer will produce better results.</p>	Maintained
	Inventor Loader	Information and data	<p>It is an extension designed to import Autodesk Inventor files. Currently only Parts (IPT) can be displayed, not assemblies (IAM) nor drawings (IDW). As Inventor files contain a complete ACIS model representation, SAT and SAB files can also be imported. Export will not be supported, neither to IPT nor to SAT.</p>	Maintained
	ImportNURBS	Information and data	<p>A workbench to add support for importing 3dm files using open rhino3dm library</p> <p>Note: This workbench is still under development</p>	Maintained
	Reporting	Information and data	<p>It adds tools to extract information from a FreeCAD document using SQL statements, and show the results in a spreadsheet. The SQL statements can be used from a graphical user interface or from Python scripts. It works in a similar way to the Arch Schedule tool but is more powerful due to the flexibility that SQL provides.</p>	Maintained
	WebTools	Information and data	<p>It contains a series of tools to communicate with web services like Git, a BIM server, and Sketchfab.</p>	Maintained
	Kerkythea	Rendering	<p>It adds a simple exporter to produce XML files for use with the Kerkythea freeware renderer.</p>	Maintained
	POV-Ray-Rendering	Rendering	<p>It creates renderings of your FreeCAD model and is very easy to use but also offers all options for advanced users.</p>	Maintained

	Render	Rendering	It can produce high-quality rendered images, using open-source external rendering engines like Pov-ray, Luxrender, and Appleseed. Render is a replacement for the Raytracing Workbench, and uses the same templates so they are compatible. In the future Render may also support Kerkythea, Blender's Eevee, and OpenCascade's CadRays engines	Maintained
	Drawing Dimensioning	Shapes	It adds dimensioning and annotation tools to the Drawing Workbench. It is deprecated since FreeCAD 0.17. Consider using TechDraw Workbench instead.	Unmaintained
	Curves	Shapes	It is a collection of tools to create and edit NURBS curves and surfaces.	Maintained
	Nurbs	Shapes	It is a collection of scripts for managing freeform surfaces and curves.	Maintained

Architectural design and FreeCAD

Arch Workbench provides FreeCAD with a modern Building Information Modelling (BIM) workflow, supporting features such as fully parametric building entities such as walls, beams, ceilings, windows, stairs, pipes, and furniture. However, to fully support the development of the project design phase, Arch Workbench needs to collaborate with other workbenches to complete the task (FreeCAD, 2021). It supports Industrial Foundation Class (IFC) files and combines with TechDraw Workbench to generate 2D floor plans. Arch Workbench imports all the Draft Workbench tools because it uses 2D objects to build 3D parametric architectural objects. The BIM feature of FreeCAD is gradually divided into Arch Workbench, which contains essential construction tools, and external BIM Workbench (FreeCAD, 2021). This BIM workbench adds a new interface layer on top of the Arch tool. The purpose is to make the BIM workflow more intuitive and user-friendly. Draft, Arch, and BIM developers also work with the OSArch community to improve architectural design using entirely free software.

Since this workbench is still under development, it lacks a level of completion of its commercial alternatives such as Revit® or ArchiCAD. However, in contrast, FreeCAD can offer a broader range of applications than this software. Arch Workbench greatly benefits from other disciplines served by FreeCAD and provides features rarely seen in traditional BIM applications.

Below listed some useful features of FreeCAD for design:

- Architectural objects are always solids. Solid objects create more error-free workflow and highly reliable Boolean operations. It becomes obvious with an operation of cutting a 3D object with a 2D plane to extract a section.
- It is possible to model architectural objects in any shape. There is no limitation for shapes. For example, walls can be placed to any degree. It is not mandatory to make them vertical. The solid object can be converted into any architectural object. Usually, it is difficult to define different

combinations in other BIM applications, like bending a floor slab on a wall, but FreeCAD does not have these restrictions.

- If the shape is too complex to model with an architectural workbench, the user can design it with the help of a different workbench like PartDesign Workbench and then convert them to architectural objects. They will still save the entire modelling history and remain fully editable.
- Arch Workbench has good compatibility with Mesh-based software.

Users can easily design architectural models in mesh-based software like Blender and import them into FreeCAD. If the model is designed with good quality and consists of multiple solid shapes, it is easy to transfer it to an architectural object.

- FreeCAD aims to create a complete material system that can define very complex materials, such as custom attributes, material families, rendering attributes, visual appearance.
- FreeCAD support IFC file format. Users can import IFC files as IfcOpenShell is installed to the application.
- Most Arch tools are still under development. For example, "wizard" tools that suppose automatically create complex geometries (for example Arch Roof, Arch Stairs) for now can only generate specific types of objects. While other tools with presets have a couple of basic presets.
- The relationship between objects in FreeCAD is not yet officially provided. These, such as the relationship between the window and its host wall, are currently implemented in Arch Workbench using temporary methods.
- FreeCAD implemented several units into the system and is planning to add more. This will make it possible for users to work with any needed unit. For now, the implementation is not complete yet.

Draft Workbench

Initially FreeCAD was created as 3D modelling Software, thus it is missing advance tools for 2D drafting. But still in recent release drafting options of the Drafting Workbench also were developed (FreeCAD, 2021). This workbench allows user to draw objects in 2D and modify the drawings. Draft Workbench can be used as a base sketcher for other workbenches such as Part or Arch Workbench, also convert draft object to sketch to be able to use for PartDesign Workbench. All this Workbenches can easily collaborate between each other. Draft Workbench has almost all useful tools that user used to have in classic CAD software for drafting – drawing (line, fillet, arch, circle, etc.), annotation (Text, Dimension, label, etc.), modification tools (move, rotate, scale, mirror, trim, etc.), among modification tools FreeCAD have such addition like – Draft to Sketch (convert draft object to sketch), slope (change the elevation of slope for draft line and wire), Shape 2D View (creates flatten 2D view of 3D object), Snap toolbar and etc. FreeCAD provide also additional features like Heal tool – deals with problematic old files, Working plane – allows the user to select the surface to draft on, can also work with arch building parts or arch section planes, Snapping –allows to place new point on objects or grid, Pattern – traditional hatch tool. Draft Workbench work with layers (must to have CAD feature). Draft Wokbench can import or export between different file formats (table).

Table 5: Draft workbench File Formats

Description	Name	Exchange
Autodesk	DXF	Import and export
Autodesk	DWG	Import and export using ODA converter
Scalable Vector Graphics	OSVG	Import and export
Open CAD format	OCA	Import and export
Airfoil Data Format	DAT	Import

BIM Workbench

BIM workbench is an external Meta workbench based on a built-in Arch workbench, initially intended as a module to gather all valuable tools for BIM projects from other workbenches and create a professional workflow. BIM workbench has specific features like wizard and management tools. BIM workbench has an intelligent tool, which is called the BIM project setup tool (FreeCAD, 2021). This tool allows the user to fill in some basic information about the project for modelling, and the tool automatically sets up a BIM project with chosen tools. It is possible to categorize BIM workbench tools in several groups: Building structure (helps to organize the model – project, Site, Building, and etcetera), BIM tools (mainly building elements like a wall, Curtainwall, Column, Window, Beam tool, and etcetera), Generic 3D tools (Profile, box, Shapebuilder, Facebinder) and native Management tools, 2D Drafting (Line, wire, arch, circle, and etcetera), Annotation (text, dimension, label, axis, and etcetera) and Modification tools (Move, Copy, Rotate, clone, and etcetera).

A management tool is a tool that organizes the project with IFC and BIM standards. Below is a given explanation for the most valuable tools (FreeCAD, 2021).

IFC elements manager - allows users to modify names, material, and IFC types of model's BIM elements. With the tool possible to change the IFC type of one or more selected elements.

IFC properties manager – this tool allows users to edit data of an object or selected several objects. This data can be attached only to the BIM object.

Classification manager - allows the user to add IFC standard class to the object or material.

BIM Windows - this tool lets users modify all openings (window, door) together with a given property.

BIM Component - allows users to turn Part-based objects to the non-parametric BIM components to export to IFC format.

BIM Preflight - allows users to check the model's compatibility with IFC standards and help reveal issues that might be fixed later.

QTO and schedule

There are several ways to extract data from model in FreeCAD. One is to use Spreadsheet Workbench. The Spreadsheet Workbench allows you to create and edit spreadsheets, use data from the spreadsheet as parameters in a model, fill the spreadsheet with data retrieved from a model, perform calculations, and export the data to other spreadsheet applications such as LibreOffice or Microsoft Excel (FreeCAD, 2021).

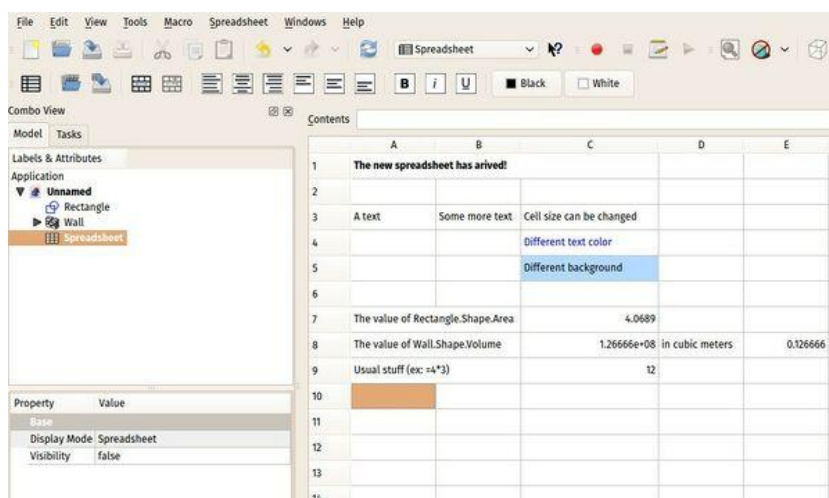


Figure 6: A spreadsheet with certain cells filled with text and quantities (FreeCAD, 2021)

The user can edit the properties of the spreadsheet. The interface is very similar to Excel, as usual, add columns and rows, control the number of columns/rows added, colour, alignment, style, unit, alias can be easily adjusted. In addition, the cells on the worksheet can contain any text or expressions. Technically, the expression must start with the equal sign '='. However, the spreadsheet tries to be smart; if a leader is missing, one will be added automatically. Spreadsheet cell data can be used in CAD model parameter expressions. Therefore, a spreadsheet can be used as a source of parameter values used throughout the model, effectively collecting values in one place. When the values in the spreadsheet change, they will propagate throughout the model. Similarly, the attributes of CAD model objects can be used in expressions in spreadsheet cells. This allows object properties such as volume or area to be used in the worksheet. If the name of the object in the CAD model changes, the change will automatically propagate to any reference in the spreadsheet expression that uses the changed name. But making changes to the spreadsheet of a complex 3D model is challenging. Editing the spreadsheet will trigger a recalculation of the 3D model, even if the change does not affect the model. For complex models, recalculation can take a long time and waiting after each edit is a waste of time. There are three solutions to this problem: Editing the spreadsheet will trigger a recalculation of the 3D model, even if the change does not affect the model. However, for complex models, recalculation may be time-consuming, and waiting after each edit is a waste of time. There are three ways to solve this problem: 1 Skip the recalculation for now. This solution has a big disadvantage. Until the document is recalculated, the new value entered in the spreadsheet will not be displayed. 2 Use the macro to

automatically skip recalculation when editing the spreadsheet. Compared with the first solution, this solution saves some steps, but it also has the above-mentioned disadvantages. 3 Put the spreadsheet in a separate file and reference the spreadsheet data from the external file. The advantage of putting the worksheet in another file instead of disabling recalculation is that the worksheet itself will be recalculated. The disadvantage is that the model does not automatically recalculate after the spreadsheet is changed. If the user first opens the "worksheet" file, change one or more values, and then open the "model" file, there will be no indication that the model needs to be recalculated. However, if both files are open, after changing the "spreadsheet" file, the "standard update" icon for the "model" file will update correctly. The sheets can be imported and exported to CSV format, and can also be read and written with most other spreadsheet applications (such as Microsoft Excel or LibreOffice). Spreadsheets in Excel "xlsx" format can be imported or opened directly from the file manager. FreeCAD checks for cyclic dependencies. By design, this verification stops at the object level of the worksheet. Therefore, users should not have a worksheet that contains cells whose values are used to specify model parameters and cells whose values use the model's output. For example, the user cannot have one cell that specifies the length, width, and height of the object, and another cell that refers to the total volume of the generated shape. This limitation can be overcome by using two spreadsheets: one is used as a data source for the input parameters of the model, and the other is used for calculations based on the resulting geometric data. When copying a cell, only the content (expression / value) is copied. Previous cell properties are not copied. Another way to extract information from the model is to use an external reporting workbench. Can be used via Python or GUI. Since FreeCAD's arc programming tool cannot flexibly extract data from multiple items, it is easy to use an external workbench to complete this task. The report module uses SQL (structured query language). SQL (Structured Query Language) is a language commonly used to manage and retrieve data from databases. But with this workbench, we can use it to select data from FreeCAD documents. This workbench does not need to install any additional software to perform all functions (FreeCAD, 2021).

Materials

FreeCAD uses the "material" property to handle the material of the object. This attribute consists of a list of values or keys composed of countless material data. However, because it is a very open and extensible way of processing such data, it also risks confusion and errors. Therefore, for each attribute, there must be a name, which is the primary key of the material. Other material properties and data are optional and can be added or extracted from the material database.

Strings separated by underscores order the names of the attributes (keys). The first substring is named after the application or standard, and the next can be used to group more attributes. Values can also be grouped by underscore. For example (FreeCAD, 2021):

- Name=Steel_Cast
- SpecificWeight=7.85 (at 20° in kg/mm³)
- EN10027_name = S235JR+AR (steel standard EN 10027-1)
- FEM_YoungsModulus = xx (in mm⁻¹·kg·s⁻²)

- FEM_YoungsModulus_Z
- FEM_YoungsModulus_X

Idea is to build up a Material Data base with the Name as a primary key. So no special data needs for material, it can be defined just with Name=Steel and FreeCAD can extract all needed data from that data base. By creating this system the idea of community is to create Data base in online and build up a general OpenSource material Library.

After creating a new material, the taskbar allows you to set various parameters.

- **Choose preset:** Select one of the pre-installed materials to be used as is or for adaptation by changing the fields below.
- **Name:** To choose name for material is mandatory
- **Edit button:** Opens the current material in the FreeCAD Material Editor, which allows the users to edit many additional properties and add their own.
- **Description:** Description of the material properties
- **Colour:** Material display colour that will be applied to all objects using that material.
- **Code:** Name and reference number of a specification system such as Masterformat or Omniclass
- **URL:** An URL for information about chosen material.

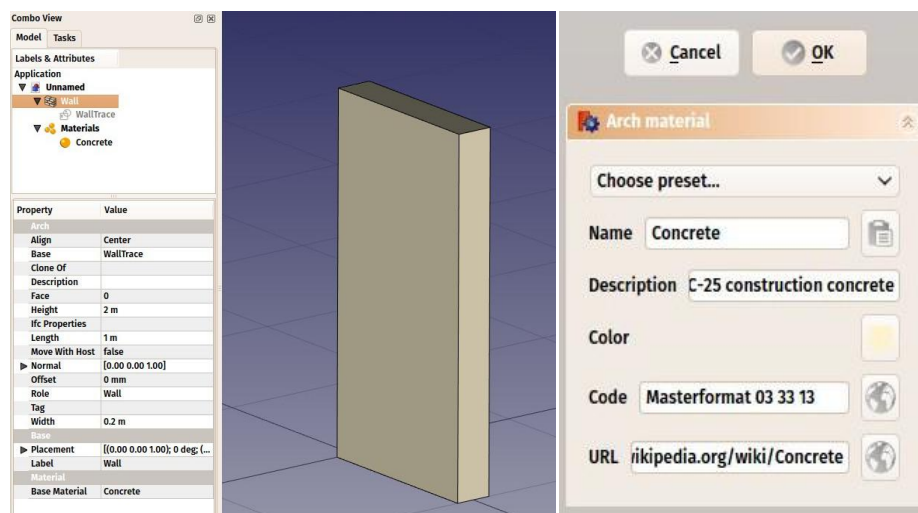


Figure 7: Arch Material properties (FreeCAD, 2021)

To be able to keep Material system organized Python front-end module was implemented and called Material.py. This module helps to calculate mass out of volume or density, to translate value to different units. This module can be implemented directly to FreeCAD or stand alone on the command

line. Ini-file format was established to be able to easily import/export material definition data. As these materials have key value form it is easy to read this format.

Each material definition is in a .FCMat file. Some of these files are part of the FreeCAD source code and are compiled into a binary file. This is done to save distribution and access overhead. But files can also be placed and searched in different places to allow for additional non-standard material definitions.

```

; last modified 1 April 2001 by John Doe

Name=Steel_Cast
Father=Steel
Source=Some material book everyone knows
(or not) ;Some comment

[EN10027]
; steel standard EN 10027-1
Name=S235JR+AR

[Graphic]
EmissiveColor = 255,255,255

```

Figure 8: Example .FCMat file (FreeCAD, 2021)

Arch Multi-Material

The Multi-Material tool of FreeCAD is very similar to object material layers of Revit®. FreeCAD use multi-material to give name and thickness value to material list and adding it to Arch object, instead of a single arch material.

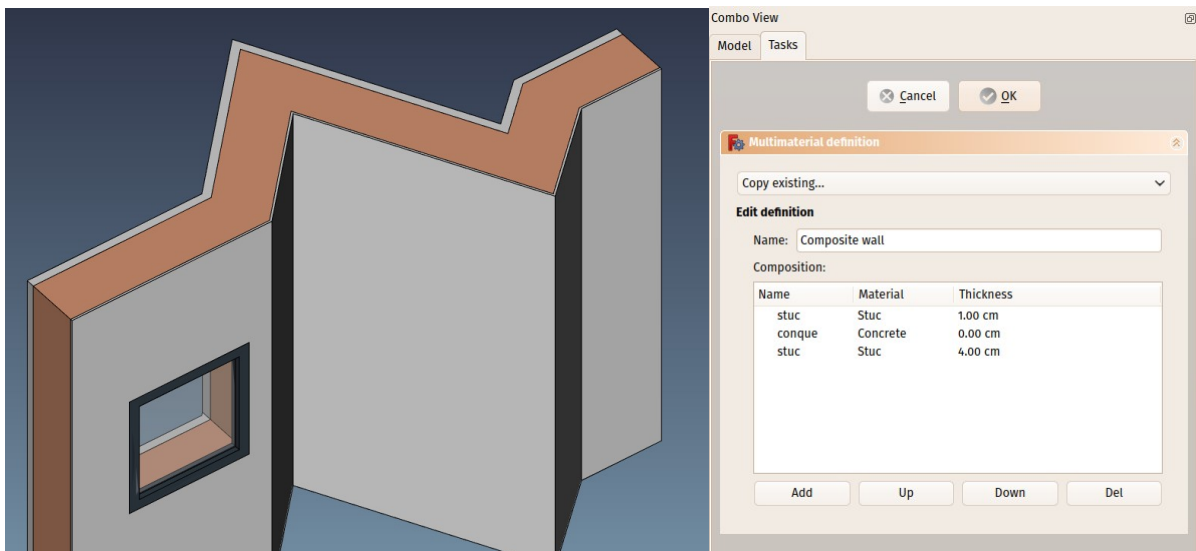


Figure 9: Multi-Material wall element and property settings (FreeCAD, 2021)

Only three Arch objects can apply a multi-materials tool, but the properties are different for all three objects:

- If to use MultiMaterial for walls, only the material definitions and thicknesses will be available to create a multi-layer wall.
- MultiMaterial for windows will attribute materials with a given name defined inside the MultiMaterial to window components with the same name or type. Material thickness is not considered.
- MultiMaterial for panels uses definitions and thicknesses of materials to create multi-layer panels.

Multi-material is editable tool. It is possible to duplicate (in this case only values are copied, and the two material don't link to each other), adding more layers to material (each layer will have its own name and thickness value), the name of material is editable (list will offer available Arch Materials in the same document), thickness is adjustable to any unit or value (Name and Material fields are mandatory, while thickness can be left as zero, in this case system reads it as variable). IFC relationship of Multi-materials corresponds to a combination of IfcMaterialLayerSet and IfcMaterialLayer (FreeCAD, 2021).

Library

FreeCAD does not have its own library. Instead, FreeCAD uses the external repository Parts Library Workbench created by the community and maintained by the community. This workbench, as all external workbenches, can be installed from the Add-on manager. Every community member can contribute their part to this repository under CC-BY 3.0 license, and each part will be copyrighted by and attributed to its author. To be able to use the library should be installed by the user, and only the BIM add-on has a library with an online option. In this case, the library can be used directly online without the need for installation. Unfortunately, the library is still under development and missing many parts that can be a problem in big architectural projects (FreeCAD, 2021).

Render and Visualization

Because FreeCAD is an application geared more towards technical modelling, it does not feature any advanced rendering tools. For this reason, FreeCAD is trying to use external engines to make good-quality renders. Currently, FreeCAD is developing an external workbench to replace the old version of the Raytracing workbench. The working idea of Render Workbench is to export a model from FreeCAD to a file with rendering format and run the renderer on that file. Now developers are trying to improve several aspects such as scene lighting, camera enhanced control, material support (Havre, 2021).

The Render Workbench is written entirely in Python, making it much easier to develop. In addition, exporters to rendering engines are implemented as plug-ins, facilitating the expansion of new engines. The Render Workbench supports Pov-Ray, Appleseed, LuxCoreRender, Blender Cycles, Intel Ospray Studio, and Pbrt v4.

It is also possible to import the FreeCAD model into Blender and run a render from Blender. Interoperability between FreeCAD and Blender is excellent; imported FreeCAD files will have materials correctly placed and ready for Cycles. Blender add-on is a FreeCAD importer for transfer files between these two software. Importing files to Blender also lets the user use Blender features for his files, such as placing textures, using other renderers supported by Blender, etc. But there is one issue with modifying files in Blender. All additions or changes made in Blender software may be lost if reimport the developed version of the file from FreeCAD. It means all changes in Blender do not affect the original model in FreeCAD. For this reason better to use Blender in the case where the model is already finished and there will be no modifications.

3.3.2. LibreCAD

LibreCAD is not parametric software and doesn't support BIM; it is very similar to AutoCAD®. Precisely this is the reason why it is on the list. Most OSS parametric software doesn't fully support 2D construction drawings and refers to LibreCAD for this task. The reason to analyze LibreCAD is to understand if this software can support other OSS parametric software with 2D drawing if needed during the project development or construction drafting.

LibreCAD is free and an open-source CAD (Computer-Aided Design) software for 2D drawing, which can handle different types of design elements in multiple industries and engineering fields. The software is free to use and develop as an offshoot of the QCAD community edition. It has a graphical user interface based on the Qt5 library, and thanks to this feature, it can run on multiple operating systems, such as Windows, Apple, and Linux (LibreCAD.org, 2020).

This project started around 2010 as a branch of QCAD 2.0.5.0. It was initially a project to build CAM functions in the QCAD community version with Mechmate CNC routers. This led to CADuntu. The project was only called CADuntu for a few months before the community decided that the name was inappropriate. After some discussions in the community and research on the existing name, CADuntu was renamed LibreCAD. Since QCAD CE is based on the deprecated Qt3 library, there was a considerable demand to move to Qt4 before further enhancements can be made. The facts have shown that porting the rendering engine to Qt4 was a difficult task, so the original LibreCAD 1.0.0 series are still based on the Qt3 support library. One of the lead developers, Rallaz, made the Qt4 migration completed during the 2.0.0 series development, and LibreCAD has become free Qt3. The latest version of LibreCAD, the 2.2.0 series, requires the Qt5 framework (LibreCAD.org, 2020).

LibreCAD may be redistributed or modified under the terms of the GNU General Public License version 2 (GPLv2) issued by the Free Software Foundation (LibreCAD.org, 2019). First, The GNU LibreDWG library had a GPLv3 license, so GPLv2 licensed LibreCAD could not use it because their licenses are incompatible. A request to relicense GNU LibreDWG as GPLv2 was also sent to the FSF, but it was rejected. This dispute has been resolved by writing a new GPLv2 licensed library called libdxfrw, and this license has more complete DWG support (Hurt, 2019).

For documentation, FreeCAD uses an online wiki page, GitHub repository, User manual source repository (Stebich, 2019). It is possible to download the User manual of LibreCAD as a PDF or access it online (LibreCAD.org, 2020). Moreover, the repository contains updated release data of

LibreCAD. Thus, documentation of LibreCAD is fully supported by the community. Unfortunately, the LibreCAD community is not as splendid as the FreeCAD community. For this reason, LibreCAD has less developed documentation, and developers are constantly posting on their pages for the need of new members.

LibreCAD uses several tools for development and scripting. These are C++ (compiler and related utilities), Boost (C++ source library), Qt (development framework), muParser (math expression parser library). The source code is hosted on GitHub and is common to all three operating systems. It can be downloaded as a "zip" file or cloned using "git" (LibreCAD.org, 2020).

LibreCAD uses DXF open format as the primary file format for import and export. Software partially supports DWG file format, only for import. The .dwg format is essential for sharing drawings between CAD programs. LibreCAD using libdxfw to help reading DWG files, but still, DWG is currently in the unfinished stage for LibreCAD but is reportedly compatible with documents saved in AutoCAD® 2007 and earlier. Except for mentioned file formats, LibreCAD also exports data to PDF and image format SVG, and other files like – ICO, JPG, PNG, DDS, TIF, BMP, but these last six formats can have bugs in the process of exporting.

The Library Browser consists of ready blocks from the categories library to download the library and insert the block into the current drawing. The insertion of a block is straightforward, just clicking on it and selecting a block from the needed categories. When inserted into the drawing, the block is shown at the base of the Block List Dock. When the block is inserted, each part of the block is converted to the block, which can be re-inserted several times. But there is one ample warning – the developers cannot guarantee that this block doesn't contain malicious code that may damage data or damage the computer system by itself. Even though these DXF files have been tested, scanned, and reworked much time in the Linux environment, still inserting these blocks are not totally safe (LibreCAD.org, 2020).

LibreCAD contains several libraries, and it is possible to specify additional libraries by defining routes to user libraries in the application settings. Also, LibreCAD organizes the library in categories by fields:

- Electronics – files inside of this library initially was created as a part of a college project in 2006 with the help of AutoCAD®, but later on, they were reworked and saved for LibreCAD, QCAD, and other FOSS.
- Architecture and Interior Design – this library mainly consists of furniture and fitting blocks.
- Electrical Engineering – blocks are containing electrical components, machines, and controls.

Interface

LibreCAD uses an interactive graphical user interface (GUI), also known as a dashboard.

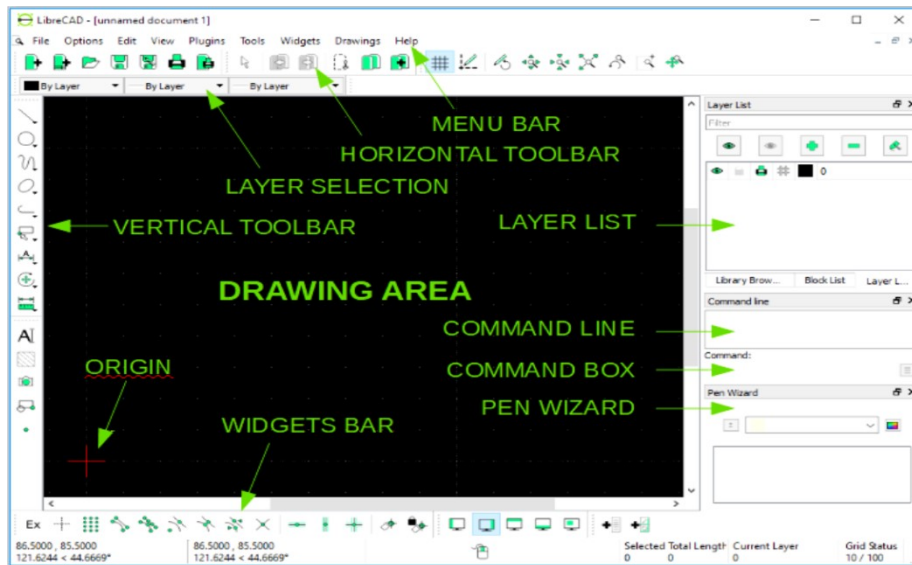


Figure 10: Interface of LibreCAD (LibreCAD.org, 2018)

The interface configuration can be different from used CAD software, but as can be seen from the figure above, the tools for drafting are very similar. LibreCAD also uses a Layer list to specify and organize design elements. The properties of the layer tool are the same as classic CAD applications, add or delete the layer, hide or change colour, width, and type of layer. Command-line works with the idea of command inputs. The complete control command list can be found in manual documentation. What is new here is the panel of Pen Wizard. This tool helps to save as a shortcut change colour of active pen, make a list of favourite colours or select all objects quickly in the same colour. The block works like a "group" command. The difference is that it is possible to save the blocks as separate files. These blocks can then be imported into other drawings. LibreCAD imports them at level 0, and they must be "moved" to another level using Modify> Attributes. In LibreCAD, it is possible to use the option to set the relative zero point (small red circle). If temporarily to set this relative zero point to a new location in a drawing, the drawn entities or newly inserted blocks will be placed to this temporary zero point.

When a new layer is created, the layer setting window appears, and it has a check box for constructing layers, which is not checked by default. Selecting it will generate a layer of helplines; they have an infinite length that will not appear in the printout. Helplines are helpful because, with their help, it is possible to outline the drawing and speed up the drawing process by adjusting the drawn lines to helplines and their intersections.

LibreCAD, like all open-source software, uses the plug-in to optimize its work. The Plug-ins bar is placed in Menu Bar on top and contains several plug-ins shown below.

Table 6: Plug-in List for LibreCAD (LibreCAD.org, 2018)

Plug-in name	Description
Align	Align the selected entity with the reference by defining the end position of the two start points

Read acsii points	Read points from a text file. Each line in the file is a point defined by ID, X coordinate, Y coordinate, Z coordinate, and optional code. Commas, tabs, or spaces can separate each field. The decimal separator is a period (.). Lines, IDs can connect these points, or coordinates, and the code field can be drawn as text.
Divide	Divide a line or circle into n parts. Markers can be placed on the boundaries of each section to show each border. The size of the tic can be defined as a percentage of the segment length. By using the Divide tool possible to break the line or circle at the boundary of each section.
ESRI Shapefile	Import GIS geospatial vector data shapefile (that is, map). Warning: The import process will block LibreCAD until it completes, and large files can take a long time.
Gear plug-in	Draw gear by selecting the center of the gear and defining parameters such as the number of teeth or modulus.
List entities	List the selected entity and its attributes, such as ID, layer, color, line type, line width, and coordinates.
Read PIC file	Import of Pic graphics language diagrams
Plot plug-in	Use the drawing coordinate system to draw mathematical functions or parametric functions. A formula, a start value, an end value, and a step value are required. The drawing can be a line, a polyline, or a spline.
Same Properties	Applies the attributes of the reference entity to the selected entity. The modified attributes are layer, colour, line type, and line width.
Sample plug-in	Draw a line specifying the X and Y coordinates of the endpoint.

3.3.3. TAD Designer

TAD (The Architect's Desktop) is a freeware software created in an architectural office by an architect to assist architectural projects from the early concept stage. Precisely this is the reason why TAD is on the list of this dissertation's study. All other software was initially made for engineering and later adopted to architecture, and as a result, they lack certain tools for architects. Although TAD is not open-source software yet, the developer announced on the OSArch official page that the next release of the software version would be open-source (Francis, 2020).

TAD started its history back in 1989. Therefore, it can be considered one of the oldest BIM applications. TAD is a product of one person. Starting from 1989, Sabu Francis, the author of it has been developing the program. Since 2017, TAD is available as freeware software (Francis, 2020).

It is an online platform with many modelling components, and all the modelled projects are saved in the cloud. To be able to use software, the user needs to register in the TAD community. The next version of the software, TAD version 7.0 is planned to be released as open-source and make it possible to install to the computer. However, here, the author is facing big problems. The application was written in out-of-date programming languages - Visual Prolog and Delphi, which are not in use anymore. Challenge is to find a modern programming language that is compatible with these old languages. Another challenge will be to change the license. After making software open-source author

will have to change it to the open-source license. Now software registered under *TAD License Version 1.2* (Francis, 2021), this license actually grants the user access to the core code of the software, changes it, or develops it. TAD has got a unique capability. The entire TAD model can be exported along with a glue code into many computer languages such as Javascript, PHP, Python, Lua, C++, C# so that one can be developed on top of it. The only requirements are if the users use modified code in commercial means, they will have to pay also to the author. The software will still be free of charge in general use but will have additional commercial offerings like private cloud servers. TAD uses an online page for documentation, but unfortunately, this page is poorly developed and missing much information on how to use the software. The only accessible tutorials are provided by the other, and they are limited. This means users will face a hard time learning how to use this software.

TAD software is Windows-based. It also runs on Linux with the help of WINE. However, as WINE is not available on Macs anymore, the application cannot be run on Mac.

Concept of TAD

TAD was created for the senior architects as an application where they could start doodling the concept of the project. In TAD, work starts from designing the space and experimenting with the shape. After the concept is decided, the 3D shape can be turned into the building objects to represent an actual project. Each object had to be given a name, and after this object gathers to the classes. Classes represent grouping in TAD. For example, all the designed space on the first floor can be added to class and name First Floor. In this case, all parts inside the first floor will be recognized as objects inside the group. The class system helps to assign common properties to the group. This tree structure or parent-child relationship in TAD is liberal. The user can create his own tree structure according to his project, while, for example, in Revit®, this relationship is predetermined by the standard classification. But after the creation of the tree structure, it is impossible to rename or delete the root of the class tree. TAD does have some amount of parameterization. But that is not like what can be seen in other software. In TAD, object "A" should be generated from object "B," so when object "B" changes, object "A" also changes (Francis, 2020).

Another interesting feature of TAD is it has a non-sequential movement. For example, the user can develop one shape, let's say number1 and after make a clone of this shape, call it number2. Continue to develop shape number2, but then decide that shape number1 can be developed in another way. Users can undo the movement on number1 while the action on number2 will be untouched even if it was developed after number1. This option is handy for the architect in the stage of concept creation. TAD also offers analytical features like calculating energy consumption or floor area, and then transfers it to the spreadsheet. This is done with the help of the scripting language ARDELA. But the result will not be construction level. TAD does not support 2D drafting, for this model should be exported to other CAD software. The model can be exported to a DXF file and then transfer to another application (Francis, 2017).

In contrast, DWG/DXF file import is very limited. IFC file format is also not supported in TAD, which means data exchange is also negligible. In addition, TAD is missing tools like curves, domes, and BRep meshes, so not all shapes possible to create in TAD (Francis, 2021).

Interface: TAD has a very simple interface to use.

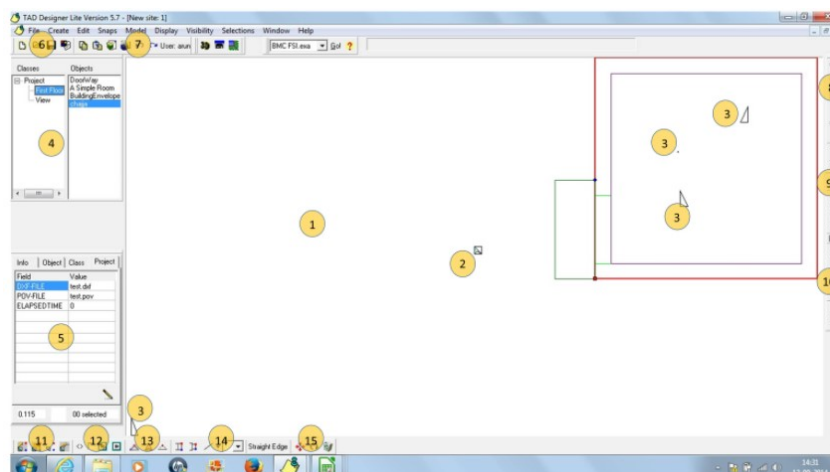


Figure 11: User interface elements (Docs, 2019). 1-Site, 2- Architect (snap), 3- modelled object, 4- Class pan, 5- Site info Pane, 6- File command bar, 7-Edit commands bar, 8- View commands bar, 9- Selection commands bar, 10- Display commands bar 11-Snap movements bar, 12- Helper movements bar 13-Vertex editing commands bar, 14- edge editing commands bar, 15-Object editing command bar

Through setting bar menu (inside File command bar) user can modify site settings by using TAD. In addition, material file users can assign material properties to the Object in TAD and set parameters for the rendering engine using Parameters for Virtualight (Francis, 2017).

Collaboration: TAD uses the "Collabalt" tool to communicate with other participants of the project. But, it is not real-life collaboration. The idea is that several architects can work on the same task and the same file simultaneously but with different sub-project files. The sub-project is a separate file within the main project file assigned to each participant. While working, each participant locks the sub-project file assigned to them. After saving the sub-project, all models will be saved to the main project file. These will merge the entire project to the same file and will be visible to all participants. Participants are not obliged to work online simultaneously; they can work offline, and the software will automatically merge the saved file to the main project file after going online. The notification will be sent to the other participant through the chat within the Collabalt tool. The maximum number of participants can be 75. But the downside of this method is a risk to overwrite someone else's work when different participants happen to modify the same sub-project (Francis, 2021).

3.3.4. BlenderBIM add-on

BlenderBIM is an add-on version originally generated from Blender software. Blender is 3D computer graphics modelling software based on the B-mesh modelling system and provides users with modelling, rendering, animation, video editing, and other simulation tools. BlenderBIM add-on also uses a B-mesh system for modelling with some extra features to adapt for BIM requirements. The main goal of this add-on is to provide tools for adaptation of project for OpenBIM and IFC standard requirements.

History: BlenderBIM was created out of demand when authors faced problem of remodelling Blender base project in Autodesk Revit®. As a solution they developed tool for Blender to support BIM and construction documentation. In October, 2019 first release was published. On February 2020, OSArch community forum was established for BlenderBIM users are able to communicate with each other and battle-testing of the application. And wiki page of OSArch changed to be documentation of BlenderBIM application development. In 2021, developers made 3 main changes and rewrite the application (Moult, 2021).

- 1- First change was to remove all the data of Blender from database and change BlenderBIM to IFC native file.
- 2- Second change was to make possible convert IfcOpenShell library to an API to simplify IFC specification.
- 3- The third madden change was to split the code of BlenderBIM into 40 modules so users can concentrate on a single module.

The results of this changes BlenderBIM community started to develop new features for structural analysis and features for scheduling and cost.

Interface: BlenderBIM uses the same interface as a Blender. Interface divided to the 4 main parts – toolbars on top and side, Area – in the middle working space, Status bar – at the bottom, property bar – in the right. Blender allows to costumize color of the screen, area of workspaces for different tasks.

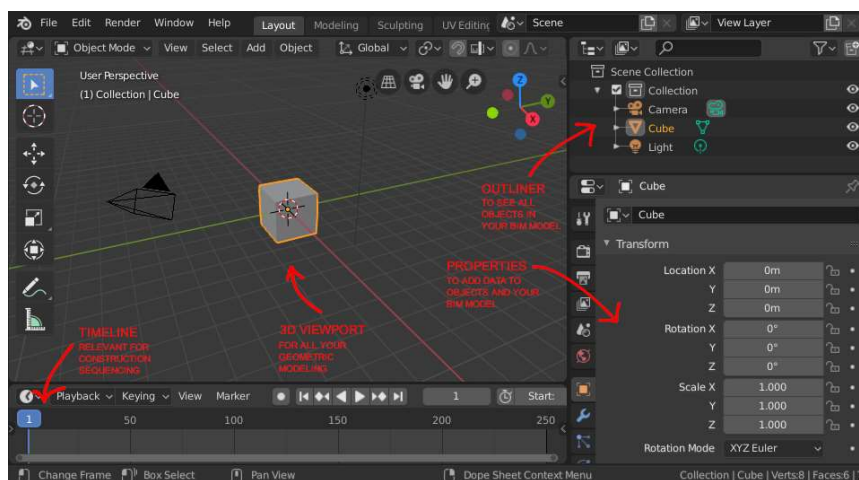


Figure 12: Blender and BlenderBIM interface (add-on, 2020)

Features: Modelling in Blender and BlenderBIM add-on has different concept that other BIM software. If other applications are object-oriented, here Blender is more traditional 3D modeller (Moult, 2021). Basically, the user models his objects in Blender and models his information layers in BlenderBIM add-on. This is very unique approach to handle BIM projects, but these way users have more control over geometry and data attached to it. However, it is important to mention that BlenderBIM is still an alpha application and can be buggy or have errors in the process of modelling. Conducted research in this chapter only focus on features of BlenderBIM add-on, as Blender alone is not modelling software for architecture and BIM field.

The main features of BlenderBIM add-on is information editing related and they are listed below (Moult, 2021):

- Import (.ifc, .ifczip, .ifcxml) and export (/ifc, .ifczip, .ifcjson) IFC related file formats
- Using MicroMVDs to check IFC data according Information Delivery Manual
- IFC data management – appoint IFC classes, giving attributes to IFC elements, attaching properties to IFC element, assigning quantities to IFC elements and calculating those quantities.
- Clash detection

Setting structure: BlenderBIM uses IFC open data structure for each project. This process in general is operated in three stages. The first stage is to create special structure of the project. The special structure follows this relationship (Moult, 2020).

- Site as IfcSite
- Building as IfcBuilding
- Storey as IfcBuildingStorey
- Space as IfcSpace

This structure creates tree structure where every element belongs to current class. User can create this structure manually or use ready setup of BlenderBIM. After, user can start modelling is project. Nesting modelled elements to the structure can be done after creating model or during the process. Any object which was not attached to any project structure will be ignored during export. The next step is to assign to each object a proper IFC class. User can assign class manually or find class using tool provided by BlenderBIM for assigning the class. This tool automatically rename and assign class to given object. Following step is adding IFC attributes to the object. The user can add attributes to the object using object property tool and pick predefined attribute from the list. Data of the attribute depend on the IFC class assigned to the selected object. Next user can add properties to the object, this properties are defined by user. They can fill the properties manually according to the object or choose from property sets. And last is the information for quantities can be added from Quantity set and they are linked to the IFC element class. This special overlay of IFC data much simple and controllable than how it is done in other BIM applications. Despite the fact that BlenderBIM is still young program, according to community members it supports IFC standards better than proprietary applications. BlenderBIM treats imported ifc format like a data that it can edit and export, rather than like a reference to read (Duncan & Filipec, 2021).

Clash Detection: BlenderBIM developed clash detection tool to examine IFC files for conflict tracking. This tool does not work directly with BlenderBIM model, to be able to conduct this task the model should be export to IFC format then reopen in Blender. This tool is very easy to use, basically, it allows user to enter code for two or more attributes and check clash detection between them. For example, if open IFC model in Blender, then write input .IfcWall in first clash source, and add input .IfcSlab is the second clash source and runs the examination, the program automatically calculates the clash between wall and slab. After, this file could be saved to the IFC file as a result. It also has an option when users exclude an element from the calculation. This means the program will calculate the

clash between all elements except the one which was excluded. This tool is straightforward and provides a very important analysis for model quality (Duncan, 2020).

Adding text, label, tags: Objects in BIM models contain a lot of information. A BlenderBIM add-on use a system of variables linked to object properties or quantities and then gets dynamically replaced with the actual related value on 2D view export. Blender mainly uses this tool to attach a tag to rooms, door and windows, fire rating, and staircase labels. There are 3 conditions to add text to the object. First the text object should be linked to the model element, the name of variable should be declared in the text object and again linked to the property of the element, and last in the name of variable should be included the text object (Moult, 2020).

To label multiple objects the best way to select all objects that will be tagged then use text tool to create text object for each object and link it automatically. Then user should select separately created text and add variables and format it.

Library: The ability to attach IFC classes and attributes to the object makes it beneficial for Blender to use ready-made 3D models from various 3D libraries that initially didn't answer the BIM standard requirements. Also, Blender can use manufacture webpage's ready 3D models; they don't require additional work like attaching IFC information to them as they already have such data.

3.4. Summary of Software analysis.

This table illustrates the detailed information for each chosen software, summarizes their features, and shows the difference and similarities between these four applications. Without question all applications has strong and weak points that they need to develop. But, as can be seen from the table more complete and more mature software in comparison with other three is FreeCAD. It has more features to support the architectural project, only weak side is it is not able to support big scale projects. Second Software which can compete with FreeCAD is Blender and BlenderBIM add-on which has better BIM requirements support. Another strong point of these applications that should be mentioned is that these two programs are compatible, and can easily exchange files among them, which will compensate weakness of each other.

Table 7: Summarize for detailed analysis of the four chosen applications

Features	FreeCAD	TAD	BlenderBIM Add-on	LibreCAD
OSS	Supported	Supported	Supported	Supported
Free	Supported	Supported	Supported	Supported
2D / 3D	Supported	Supported	Supported	only 2D
Parametric	Supported	OOPs	Supported	Not Supported
Visualization	Raytracing Workbench Render Workbench	plug-in	Blender built-in Eevee, Cycles	Not Supported
links to external	Blender		Octana, Redshift,	Not Supported

visualization	POV-Ray Lexrender	Renderman, POVray	Radeon, Prorender	
Documentation	FreeCAD Wiki Page Online Manual	Tad Doc Page	WikiOSArch page	Online Manual , Wiki page and PDF
Construction drawing/drafting	Draft Workbench Sketcher Workbench	Not Supported (should export to regular CAD)	Construction lines plug-in	Supported
QTO/ construction estimate	SpreadSheet workbench	Ardela	Supported	Not Supported
Open object Library	FreeCAD Parts library		Supported	Library Browser
Place image	Image workbench	Supported	Supported	Supported
import / export schedule	Arch Schedule tool Reporting Workbench	Supported	Built in Schedule	Not Supported
Topography	Arc Site Trails Workbench	Not Supported	Supported	Not Supported
Support point cloud	Points Workbench	Not Supported	Supported	Not Supported
IFC import/export	IFC	Not Supported	Supported	Not Supported
Clash Detection	Not Supported	Not Supported	Supported	Not Supported
Layered object (complex object)	Multi Material tool	Not Supported	Not Supported	Only Layers for drawings.
Tutorials	Manual, External sites, YouTube	Youtube, official Webpage	Youtube, WikiOSArch	Online Manual, built in Tutorials
Installation	Windows, Linux, Apple	Windows, Linux	Windows, Linux, Apple	Windows, Linux, Apple

4. SURVEY ANALYSIS

Survey analysis for this dissertation was done by method of collecting data through a survey among architectural design professionals and software community members. Survey has two parts:

- 1- Survey for the first part of the case study was conducted among open-source software community members about the comparison of proprietary software with open-source applications, their opinion and experiences with modelling and the usage of open-source programs. Web pages used for this survey are FreeCAD forum, LibreCAD forum, OSArch forum.
- 2- The second part of case study consists of questionnaire in Google form sent to AEC scope professionals to gather information about their knowledge and experience with BIM and perception about open-source software. Survey was published in LinkedIn webpage and asked AEC professionals to fill it. Additionally, it was sent to architectural companies and asked their opinion about OSS.

4.1. Comparison of Proprietary and open-source Software

For the purpose of evaluating open-source software performance, the case study focused on comparing proprietary and open-source software. This dissertation compares open-source projects like LibreCAD with classic AutoCAD®, BlenderBIM and FreeCAD to Revit®, the dominant proprietary application in the market. Community members of each software actively and eagerly participated in the discussions and shared their experiences.

4.1.1. First Case: LibreCAD and AutoCAD®

Pricing and License: LibreCAD is open-source software and free to use for everyone. AutoCAD® is one of the oldest and classic CAD software. But it is proprietary software that belongs to Autodesk, and as with all products of this company, this application also requires payment for use. However, a 1-month trial is free for use, and the student version does not require payment for one year.

Features: LibreCAD and AutoCAD® have similar tools for drafting. It is almost equal in drafting capacity. Both applications work with drawing tools, layers, layouts, entity modification, hatches, block management systems, annotations, units, and dimensions tools. In addition, both have customization for toolbars layout, the available user interface in different languages, and coordination systems. The only thing that LibreCAD is missing is the 3D CAD that AutoCAD® features. But LibreCAD supports isometric drawing, which can be convenient for mechanical part drawings. Another problem of LibreCAD is when offsetting polyline with fillet tool software or giving inaccurate results. In general, if comparing drafting speed, most users agreed that LibreCAD requires significantly more time on draft, edit, and document than AutoCAD®.

Flexibility: In terms of flexibility, of course, LibreCAD wins the race. LibreCAD is an open-source application, which supports plug-ins, and in this term, is much more flexible than his opponent. While

AutoCAD® is frigid with his abilities of drawing. Plus, being open-source means, it can be developed according to the demand of the user.

Native file format: AutoCAD® works with DWG (closed proprietary format), a basic file format for data exchange that existed long period of time. Most of the software can read DWG files, so LibreCAD also can import DWG files. For the LibreCAD native file, the format is DXF (open source format) generated from AutoCAD® and compatible with almost all design software. The biggest problem of LibreCAD is it cannot handle large files (urban scale project, landscape project scale), the software or crashes, or don't read all information on resorted in the file. AutoCAD® doesn't have a problem with reading large files, and the problem is not always it can handle heavy drawings. But it does not guaranty that AutoCAD® does not crash or freeze.

Layout: Here is the main problem of LibreCAD. Most users indicate that they are highly unsatisfied with the layout solution and proper referencing of LibreCAD, as it is one simple template. While AutoCAD® manages this option very professionally and very well with documentation.

Conclusion: LibreCAD is good mature 2D software for small architectural companies which don't participate in huge construction projects. As LibreCAD lack the speed and ability to support heavy files, it will be hard for a firm to deliver good quality drawing on time for the construction. LibreCAD was compared to AutoCAD® to understand if this software can support other BIM open-source software with 2D documentation if the company deals with big and complex architectural projects. None of the applications is object-oriented and none of them is BIM-capable software. They can only support the project with specific drawing task with limited metadata.

4.1.2. Second Case: FreeCAD and Revit®

Pricing and License: FreeCAD is a totally free open-source application. Users are not obliged to pay for using the software; only some external plug-ins or add-ons can ask payment. In contrast, Revit® is free to use only for trial 30 days and free for one year of use for students. For professional service, Revit® requires monthly or annual payment upon registration.

Native file formats: The native file format for FreeCAD is .FCStd (open source format) extension. Blender can open this format with the use of plug-ins. FreeCAD does not have restrictions for opening native file format in the older or newer software version. The only limitation is when the complex object is created in a recent version, it might not be editable in the previous or older version. Native file format for Revit® is a .rvt extension (closed proprietary format), which can be open only by Revit®. A newer version of Revit® opens a native file from an older version. But the opposite process is not possible; the older version of the application will not open files coming from the new version. This limitation forces the user to update the software version constantly.

Interoperability between BIM models and data exchange: Revit® was initially created as BIM software; it has exchanging abilities with IFC, DXF, DWG, and BCF files (with plug-in). While for FreeCAD interchange is still limited. FreeCAD is trying to improve the compatibility issue with the IFC format, but for now, exporting data to the IFC format is not fully supported for all objects. Although FreeCAD supports STEP file for import and export. DWG file exchangeability is possible

only with the help of an ODA plug-in. DXF file is well-supported in FreeCAD, but it only can be used for 2D exchange.

Project Units: FreeCAD support all possible units for modelling. Revit® chooses the template before starting the project and lets users to modify units from “manage property”. In Both applications changing the unit does not affect the model.

Element organization: FreeCAD uses an advanced grouping option to organize modelled elements. For example, the user can group building elements in different levels, storey, building parts, or other FreeCAD workbenches. While in Revit®, modelled objects are automatically organized into Floor Plans and, after possible, reorganize objects to groups.

Object Family: Revit® is object oriented software and uses the family relationship of objects to organize the work. Each object is a part of the corresponding family and all the objects are parametric. FreeCAD has a different concept. It uses the idea of cloning with the adoption of the shape and properties of a cloned object. Contrary to the family concept that Revit® uses, the modelling is achieved by cloning a related object which in turn copies properties of the object it turns into. Another method that can be used is to construct a similar structure on the same base profile and make it possible to change the size of objects, which makes the object partially parametric.

BIM tools: FreeCAD and Revit® have approximately similar tools for BIM modelling, which are walls, structural elements (beams, columns, slabs.), openings (door, window), roof, stairs, frames, and equipment. But the difference is on the development level and applicability of this element. Revit® has more high-level use of this tool with different variations, while FreeCAD is still developing BIM tools, and for now, usage is still limited in shapes and methods.

Modelling: Revit® tools are more refined than FreeCAD tools, considering how much money private companies spend for Research and Development to develop this type of software. Revit® has pre-made wall families that can be easily combined and moved. The layering system lets users design new walls, insert each layer's properties and material, and create a new object within a family. The placement of the opening tools, doors, and windows are also proficient and fast. When placing the door or window system, the system automatically calculates and gives an opening to place the object. If the object is deleted, the system logically brings back the wall (below this process explained with example). One of the advantages of Revit® is 2D drawing, construction documentation, annotations, and sizing. This is the most significant deficiency of FreeCAD in architecture: it is difficult to make 2D drawings with hatching in larger projects. It is not impossible, but rather very time-consuming and requires more technical knowledge and the use of external tools. The best choice to create professional 2D drawings is to export the drawing to QCAD or LibreCAD, as there are more professional drafting open-source applications. The most significant advantage of FreeCAD is the liberty of modelling. The user is free to model any possible shape or figure, plus FreeCAD does not have a preset for objects to be modelled. However, this liberty sometimes can be problematic for FreeCAD if the scale of the project is big, large-scale projects can be too heavy if modelled in FreeCAD, and software can freeze from time to time, especially with the shape 2D view option. Below there are four examples of 4 BIM tools in use. These tools are – wall, slab, window, and stair.

Modelling wall: Revit® has a very intelligent system with dependent objects. Assigned properties of one object directly affect the depending model. For example, modelled walls' height adapts to the height of the floor level that was initially set. If the height of the floor level is altered during the process, the height of the walls changes accordingly. Revit® also provided a simple mechanism to override this behaviour, allowing some walls to end below or above the desired level. Height modifications are entirely up to the user in FreeCAD. FreeCAD grouped walls together in a way that made manipulating member walls difficult. After modelling an area with four walls, clicking or picking any of the walls selected the rest of the chain. Another problem with drawing walls is when starting to model them from the line, each line should be chosen separately and then apply wall button and then repeat this process for every line. In the end, choose all walls together and give them the height. This process is time-consuming and not adequate.

Modelling Slab: Revit® and FreeCAD have the same approach for creating slab elements. Both first ask the user to create the 2D perimeter of the shape and then extrude the 3D slab object. But Revit® keeps the thickness and boundaries of the slab adjustable. In contrast, FreeCAD cannot do it in one process. Arch workbench expects the user to create the profile of modelling slab in Sketch workbench and then import it to arch workbench and use tool slab to create a volume. Otherwise, the Arch workbench model uses a default structural element. However, importing from Sketch workbench also does not guarantee success. Initially, this workbench was not planned for architectural modelling to work with complex sketch shapes, which can be problematic for Arch workbench. If developing the slab model further, the user can face another problem creating holes in the slab for stairs or atrium openings. Revit® creates these voids automatically according to the function, while FreeCAD uses Boolean operation to create them. This means it will need to create a secondary object to be able to use the Boolean tool. This process is time-consuming in comparison to the straightforward approach of Revit®. But unlike Revit®, FreeCAD let the user define created element as a different structural object (slab, beam or column), while Revit® is strict with functional belonging. This is a very logical approach, as most of the structural elements depend on profile creation.

Modelling stairs: Both applications have a similar approach to creating stairs. In Revit®, the designed stairs still have adjustable properties; if needed, they can be easily modified for new dimensions. On the other hand, the Arch workbench of FreeCAD provides a well-detailed parametric stair to use, but it is only a straight stair; any different shape should be modelled using extra workbenches. Plus, the properties are not as flexible as the Revit® model, it is limited for modification by properties panel.

Modelling window: Here, the logic is the same in both applications: Picking window model and inserting it to the wall. Unlike Revit®, which offers an extensive selection of window and door objects, FreeCAD only has a small amount of window models, plus door consider being a special case of window. For more window types the users should switch to the Sketch workbench, create 2D shapes with constraints associated with the user interface, letting the user modify the values of the shape. Being able to change the value of the shape accordingly means an update in form. After that, the user adds thickness to the given shape in the Arch workbench to create 3D geometry of the object. BIM windows management tool of the external workbench let the user change properties like height or width of several windows at the same time. This is very satisfactory development of FreeCAD from the previous version, where this process was done manually.

Conclusion: This study shows that FreeCAD is still missing some intelligible consequence design approaches. It still keeps a minimalistic method in the creation of AEC objects. The process was time-consuming and technique-oriented, with a lack of classification guiding rules. FreeCAD is good with the freedom of creating complex shapes and objects, but it lacks an organizational approach. Another problem of FreeCAD it is hard to tell which direction software aims to develop. One of the Forum members said FreeCAD is more of a bazaar than a cathedral. Considering different workbenches, one can say that software development goes in many different areas without having one central organization. It is understandable that all workbenches were born out of the necessity of different users, which is why some workbenches have a hard time with the lack of tools. The diversity of FreeCAD also can be its weak point. The decentralized development of FreeCAD also affects the improvement of the IFC data exchange, as developers cannot enhance the interoperability with IFC, because software changes very rapidly in different direction. Also, the lack of control metadata on the model slows down the process. This, in turn, hinders the compatibility of FreeCAD with BIM systems.

4.1.3. Third Case: BlenderBIM add-on and Revit®.

Pricing: As a FreeCAD, Blender is also totally free software, which doesn't require any payment for the usage of the application.

Modelling: It is not correct to compare these two applications in terms of modelling, as they have totally different modelling concepts. Revit® is object-oriented software, while Blender uses a B-mesh system for modelling. One way of modelling in Blender is basic modelling methods like the morphing method and curve method. The difference between these two methods is that the morphing method is editing basic objects like cylinder or cube or sphere, extruding them, cutting or sculpting geometry, etc. The curves method has a three-dimensional geometry creation based on curves moving around another curve or combining several curves into the surface. This allows the user to create curved shapes or topographic surfaces.

Another difference between these two computer programs is they have different parametric designs. Revit® is a fully parametric program, where this process is totally atomized by software. While in Blender, parameterization is controllable. The user can assign constraints to the model or create parameters to properties to control the object modification. For example, to control repetitive copy modification, the user can add parametric modifier to properties that will control the modification of the copying. Another method is to create a script to control parameters. But the user should consider creating these tools, and assigning them to the object is a manual process, which requires professional knowledge about modelling in this software.

Like Revit® with Dynamo, Blender also has a tool for computational design, Sverchok, which is also open-source. But this add-on is very new and not developed enough to be reliable.

Drawings and documentation: The weakest point of the BlenderBIM add-on is 2D drawing and documentation; it has minimal tools for this task which are still in the initial development stage. In contrast, Revit® has advanced capacity to produce high-level construction documentation and provide the project with quality drawings. To improve its performance, BlenderBIM should use additional plug-ins for this task. Construction lines are paid plug-in for Blender to produce CAD style modelling.

But this add-on is a Beta version and still underdeveloped and constantly updated for bug fixes. As Blender and BlenderBIM focus on different goals, they don't give access to highly specialized Drawings tools, making it impossible to produce professional project documentation.

Interoperability between BIM models and data exchange: In this question, Blender has more credits than Revit®. BlenderBIM add-on directly read and edit IFC file, it treats IFC standard as native way of recording information about the model. However, a special overlay allows describing objects in accordance with the IFC standard. The implementation of the IFC standard in the case of solutions presented by Blender and FreeCAD is much simpler than in the case of other BIM programs. Support for the IFC standard, despite the short history of BlenderBIM, for example compared to Autodesk Revit®, is much wider. It does not require creating new tools in the program or objects, but it comes down to introducing new IFC classes. At the same time, it should be noted that the discussed software still requires the project to be converted to the .ifc format, however, in this case it is only related to the translation of data into and from the Blender language.

In contrast, Revit® treats IFC files very poorly. It takes several seconds to import small IFC files, but the waiting time for the import of such files in Revit® is multiplied by more than 10 times. In order to carry out a simple filtering operation or read the file, Revit® maps it and creates another Revit® file and prohibits the file from being modified. This creates several obstacles, such as the usage of the intellectual properties by the third party, even if it is allowed by the owner, and sometimes even the owner cannot access his or her own file. Moreover, mapping of system families is impossible in Revit®. A famous example of Revit®'s treatment of IFC files among the community members is exporting two identical elements (one being the clone of another) to .IFC files. Revit® classified one element as roof, while the other one was classified as a slab. The algorithmic logic behind Revit®'s classification process is unknown (Kowalczyk, 2017).

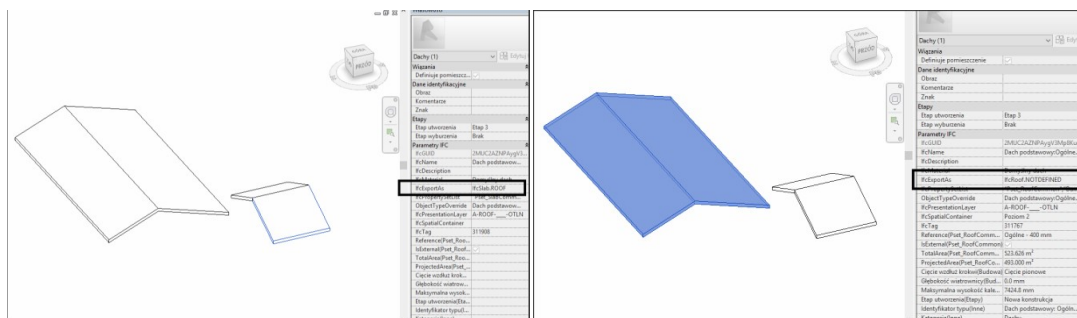


Figure13: IFC classification mistake in Revit. (Kowalczyk, 2017)

Revit®'s another notable problem is the issue with the Omniclass specification of objects. The Omniclass specification file that comes with the Revit® application is outdated and therefore, does not let system families to be classified.

Revit®'s treatment of IFC classifications is another issue as it lacks clarity. The IFC classification option is not visible in the program's interface. Instead, this feature is kept separately from the main window. Therefore, without knowing exactly where to look, the IFC classification process may fetch negative results.

Below shown summarized table of comparison between applications analyzed in this chapter.

Table 8: Summary of comparison between chosen applications

Criteria	AutoCAD	LibreCAD	FreeCAD	BlenderBIM	Revit
Pricing	Payment required	Free	Free	Free	Payment required
Modelling	2D and 3D	Only 2D	Partially parametric, 2D, 3D	Partially parametric, 2D, 3D	Parametric, 2D, 3D
Algorithmic modeling	Not supported	Not supported	Supported (FreeCADMacro)	Supported (Sverchok)	Supported (Dynamo)
Object Oriented	Not supported	Not supported	Partially supported	Not supported	Supported
Customization	Partially Supported	Supported	Supported	Supported	Partially Supported
IFC support	Not supported	Not supported	Partially Supported, under development	Supported, full control over IFC exchange	Supported, Limited control.
Plug-ins	Not supported	Supported	Supported	Supported	Supported
BIM tools	Not supported	Not supported	Supported	Supported	Supported

4.2. Survey about Open-source software with professionals in AEC sector

The survey was done in order to measure the knowledge of professionals about open-source software, and the place of open-source software in the Architectural software market. Moreover, it was essential to know which criteria specialists preferred when choosing software to work with. The survey will help to examine the availability of modern architectural companies in adopting open-source software solutions.

The survey is divided into two parts. The first part of the survey is dedicated to general questions about BIM, users' experiences with it, and its workplace usage. The second part involves questions about the open-source applications and the respondents' knowledge about them. The number of participants amounted to 45. The survey was prepared using Google forms and was published on LinkedIn platform and AEC professionals were asked to fill it. Additionally, it was sent to architectural companies to receive their opinions regarding the OSS. Only the most notable pairs of answers are shown in the graphs. To be able to calculate result more accurately the question with multi-choice answers were shown in percentage.

The questions and graphs for the first part of the survey are shown below. The questions are about the professional occupations of the participants, their knowledge about BIM and whether their company uses the services of BIM software. Moreover, they were asked about the duration of their experience with BIM to have a general understanding about their readiness to adapt to new technologies.

Table 9: Survey questions about BIM awareness in AEC companies

1	What is your professional status in the Architectural field?
2	Do you have knowledge about BIM and BIM tools?
3	Does your company use BIM? If not, does the company plan to implement BIM?
4	How many years of experience do you have in using BIM?

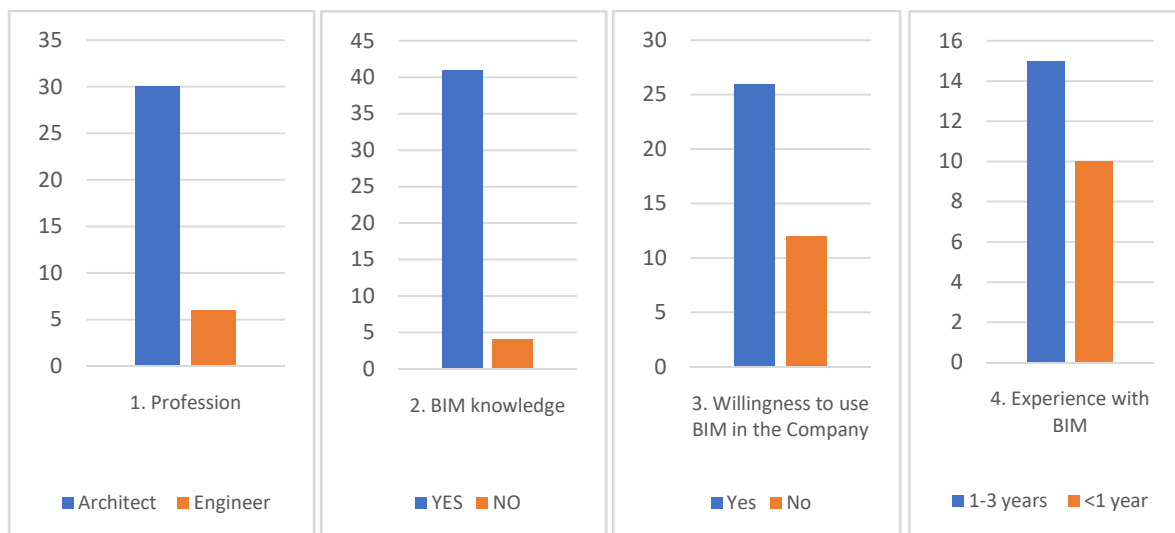


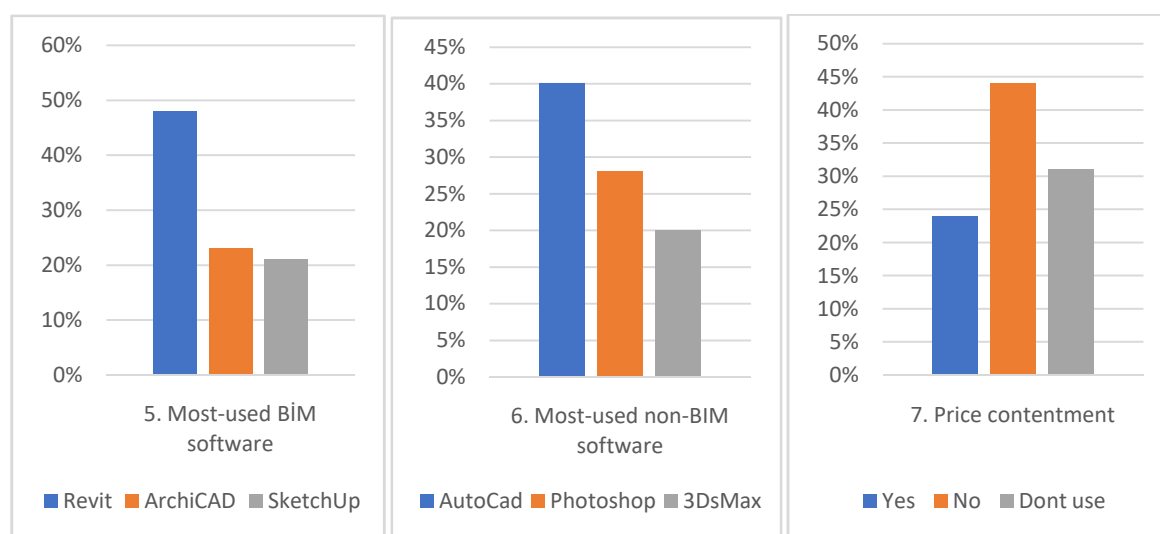
Figure 14: Chart showing BIM awareness among AEC companies and professionals

30 of the respondents were architects, while engineers were the second most represented with 6. An overwhelming portion of the respondents (41 of them) claimed that they knew what BIM is. But although most of them were aware of the existence of BIM, only 65% of them used BIM at the corporate level, meaning that only 26 of them used BIM applications professionally.

When it comes to the period of usage of BIM, one-thirds of the respondents stated that they had between 1-3 years of experience with BIM, which was the most frequently given answer. On the other hand, 10 of them were relatively new in the BIM sphere, with less than one year of experience, while the most experienced (5+ years) users occupied third place with 16%. Users with 3-5 years of experience amounted to 12,9%, the same with the inexperienced ones. These results show that BIM has only started to be relevant in the working sphere in the last 3 years.

Table 10: Survey questions about Software used by AEC professionals.

5	Which BIM authoring software does your company use?						
ArchiCAD	Revit	Allplan	Bentley	Edificius	Sketchup	BrisCAD	Other
6	Which other software does your company use?						
AutoCAD	Photoshop	3ds Max	Microstation				Other
7	Are you content with the financial policies?						

**Figure 15: Chart show software choice among AEC professionals**

According to the respondents, the most-used BIM authoring software is Revit®, with 48% of the companies the users work for utilizing it. The second most used application is ArchiCAD, with 23% companies using it, and the third one in the list is Sketchup, with 21%. Among the non-BIM applications used by companies the participants work for, AutoCAD® is the leader, with 40% of the users stating that their companies use it. Photoshop was the second with 28%, while 3DsMax stood at 20%. Meanwhile, Rhino's usage percentage was 6%, same with the other programs mentioned in the survey.

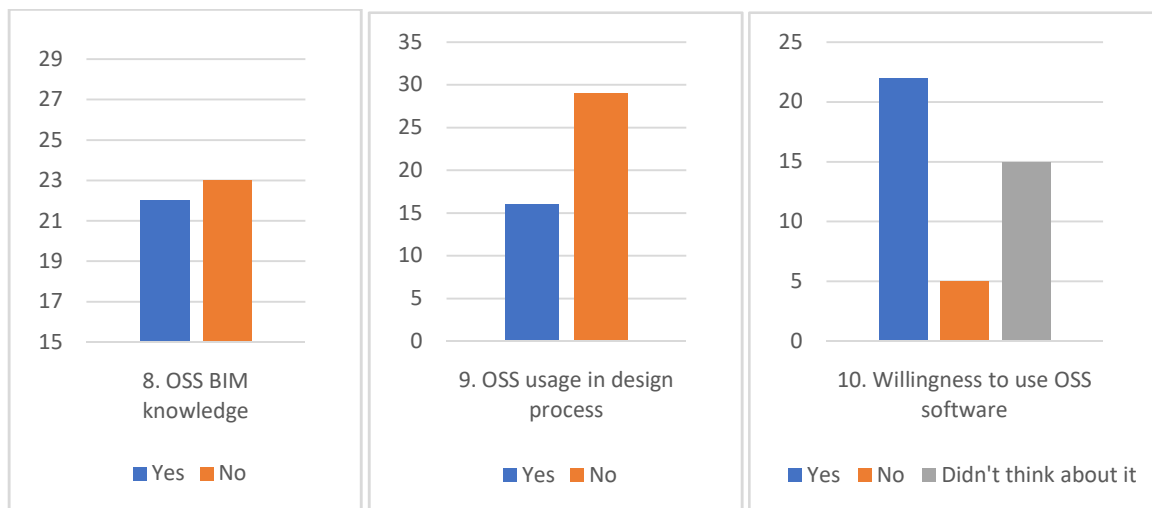
Since proprietary software requires licenses that cost money, it was essential to ask whether the users were content with their pricing. In this respect, most users responded negatively to the question, as the licenses' prices were too much for 65% of them.

The answers indicate that the professionals still prefer private tech companies. And although the prices of proprietary software do not satisfy the user demand; they still have to pay the money, as there are obliged if using proprietary applications.

Additionally, participants were enquired about the existence of Open Source software and whether they use it in their professional endeavours.

Table 11: Survey questions about OSS awareness among AEC professionals

8	Do you have knowledge about Open Source BIM software (software with open shared code)?
9	Do you use Open-source Software in your design process?
10	Would you like your company to use more Open source BIM Software?

**Figure 16: Chart shows OSS awareness among AEC professionals**

When asked whether they knew about Open Source BIM software, about half of the respondents (22) answered positively, the other half, however, claimed that they did not know anything about it. Moreover, when asked whether they used open-source software in their design process, most users (64%) had a negative response. Almost half of the respondents (22) claimed that they wanted their employers to use open source BIM software, while 15 of them answered that they had never thought about it. Only 5 users claimed they did not want such a thing to happen.

The result is most of the users and companies in the market are unaware of the open-source software and the benefits of using such applications. As a result of the lack of knowledge about the alternatives, they are choosing to use proprietary software for their projects.

Furthermore, participants answered questions about which criteria they valued in the software they stand to use.

11. Which criteria are you considering in choosing the software for your design?

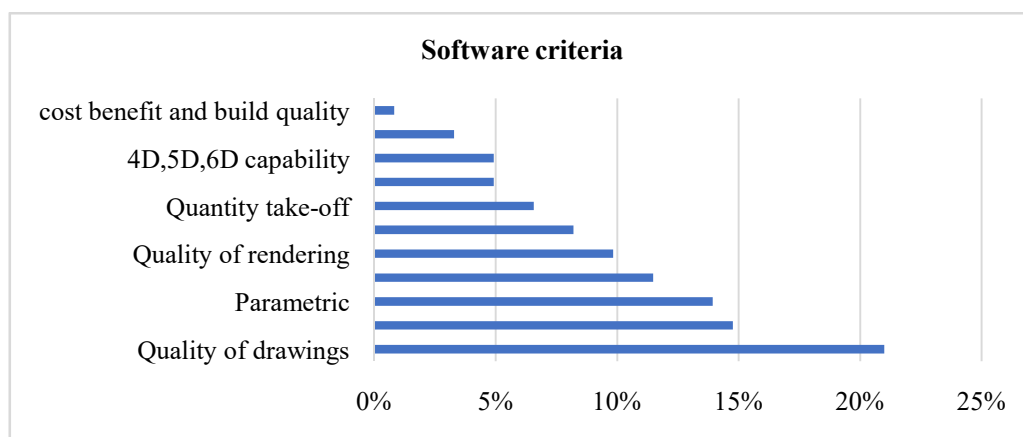


Figure 17: Software criteria preferability chart.

The crucial criteria for software users when choosing Architectural design programs is drawing quality, with 21% of the respondents opting for it. Document management also received a considerable number of votes, with 15% of the respondents opting for it. Parametric is also essential for users, documented by 17 votes which translates into 14%, while Project management (11%) and quality of rendering (10%) are also factors to consider when choosing a suitable application.

Participants were asked about open-source software they knew or use:

(12. Which Open-Source Software do you know?). Most of them answered with Blender, the rest mentioned proprietary software names, which means professionals in AEC field are not knowledgeable about the origin of applications and don't differentiate between open source and private software. When asked about which other criteria people pay attention to when choosing software to work with (13. Which other software criteria do you pay attention when choosing software to work with?), most of the participants mentioned interoperability, clear interface, usage without the installation of additional plug-ins, and price.

Overall, it is clear that professionals have a strong interest and willingness to incorporate BIM and especially, Open Source BIM into their works, but companies still keep using proprietary software and do not rush in making a transition. Professionals are unfamiliar with these applications and what benefits they can have by using them. Also, the lack of consistent support for OSS creates mistrust which hinders the process of adoption. Another reason for professionals' hesitation regarding the usage of open-source applications is the lack of certain functions and the dependence on add-ons or scripting. Most engineers and architects lack scripting skills and it makes it hard for them to work with OSS. The lack of resources or limited number of tutorials showing how to use open-source applications also plays its role in avoiding choosing them to adapt to AEC companies.

5. CONCLUSIONS

The main question that this dissertation revolves around is the following: Are today's Open-Source software solutions advanced enough to support architects and architectural companies, as well as designers in the project design phase? Do they have enough capability to perform at the same level as their proprietary alternatives? It was also essential to understand the extent of the awareness of professionals in the AEC field regarding the OSS software and its potential benefits.

The importance of BIM is increasing, so the demand for capable and acceptable software that can answer the current requirements of BIM. This demand has encouraged developers to come up with adequate solutions that can be transparent and more open for. Recently, more and more professionals are choosing to develop OSS to be able to control the process of design. The investigation was conducted by using three models: 1) Understanding the capacity of OSS software in the field of architecture, 2) Doing a survey to learn the opinion of professionals about OSS and how it is applied in architectural software, 3) making an enquiry among OSS community members to understand how OSS software compares to its proprietary counterparts. Since they have experience with both types of software, their knowledge and personal thoughts may reveal the answer to this question.

For this dissertation one can summarise the following conclusions:

- 1) This study shows that although they may prove to be effective in small-scale projects, Open-source applications have not yet reached the level of maturity the proprietary programs show when it comes to managing multi-scale architectural projects as they lack a lot of features that aim to solve specific tasks that come with big-scale projects. Most of them depend on plug-ins to fetch desired results. This solution is not convenient for architects as they lack the knowledge of scripting skills that are necessary to run those additions.

As it offers more freedom and flexibility, open-source software has a huge potential, as the development of such software is triggered by the demand of market and professionals. But sometimes, this flexibility may lead to the lack of centralization, as they start growing in many unconnected directions and may eventually halt the influence of that application.

- 2) According to professionals, OSS is a topic that is worthy of a consideration. But because of its free nature, it gets little attention. Most of the Open-source applications are unknown, although they have enormous and virtually limitless potential. Respondents did not know much about OSS, although those who were acquainted with OSS and its capabilities were willing to give it a try, and their satisfaction with the functions of the open-source software translated into their willingness to try it in the corporate level. Being free, open to almost all kinds of modifications, and manageable makes OSS a potentially money-saving option for architectural companies but as the survey showed, firms still take their time and wait for the developments in order to make a decision.
- 3) The research shows that both open-source and proprietary software has its strengths and weaknesses in comparison to each other. In terms of modelling, open-source software is advantageous. Because of the flexibility that it offers to the user. It offers a considerable

amount of freedom to the user thanks to plug-ins that can be installed or scripted, and used to create almost any shape that is needed. In contrast, proprietary software is more practical and result-oriented and consequently, less flexible for modelling. Proprietary software was created as a tool for designing more conventional projects and provides users with more standard packages of tools.

Overall, it is safe to say that while each type of software has its pros and cons, eventually it depends on the needs or demands of the individuals use them. It depends on the style of architects and their architectural approaches. It seems, for more experimental professionals, open-source programs create a large opportunity to realize their ideas. For more conventional styles, however, it is possible to choose the software with more conservative approach and tools.

Although there are benefits of utilizing the abilities of computers for design, there also exist limitations in collaboration. Dominant applications in the market create closed environment as their codes are private and this halts the process of interoperability among participants of projects, as they cannot modify software exchangeability according to the need. In order to work more closely-knitted and with more freedom without any artificial obstacles on their way, OpenBIM solutions should be integrated into the process of design. Open-source applications are more compatible with OpenBIM ideology and this fact makes them more efficient for architectural companies to work with. Companies will be less limited with software and interoperability among platforms will be of satisfactory level.

BIBLIOGRAPHY

- Aaby, A., 2004. *Introduction to Programming Languages*. Washington.
- add-on, B., 2020. *BlenderBIM beginners tutorial: my first BIM project*. [Online] Available at: <https://blenderbim.org/blenderbim-tutorial.html> [Accessed 04 September 2021].
- Ankerholz, A., 2020. *Open Source Software Licensing*. [Online] Available at: <https://www.fossilife.org/open-source-software-licensing> [Accessed 01 August 2021].
- Anon., 2020. *BIM application compatibility table*. [Online] Available at: https://wiki.freecadweb.org/BIM_application_compatibility_table [Accessed 06 September 2021].
- Anon., 2021. *File Format FCStd*. [Online] Available at: https://wiki.freecadweb.org/File_Format_FCStd [Accessed 24 Augustus 2021].
- Anon., 2021. *FreeCAD setting up a model for IFC export*. [Online] Available at: https://wiki.osarch.org/index.php?title=FreeCAD_setting_up_a_model_for_IFC_export [Accessed 07 September 2021].
- Baldwin, M., 2017. *What is openBIM*. [Online] Available at: <https://bimconnect.org/en/wiki/what-is-openbim/> [Accessed 11 August 2021].
- Berman, D., 2020. *Open Source Licenses: Types and Comparison*. [Online] Available at: <https://snyk.io/learn/open-source-licenses/>.
- Botsman, R. & Rogers, R., 2010. *What's Mine Is Yours: The Rise of Collaborative Consumption*. New York : HarperCollins.
- Bretthauer, D., 2001. *Open Source Software: A History. Published Works.7.*
- Carillo, K. & Okoli, C., 2008. *The Open Source Movement: A Revolution in Software Development*. Montreal.
- Costa, C.D., 2020. *Best Python Libraries for Every Python Developer*. [Online] Available at: <https://towardsdatascience.com/best-python-libraries-for-every-python-developer-77daab4fa40e> [Accessed 05 August 2021].
- Davies, J., 2018. *Building great open source documentation*. [Online] Available at: <https://opensource.googleblog.com/2018/10/building-great-open-source-documentation.html>.
- Docs, T., 2019. *Understanding the User Interface*. [Online] Available at: http://docs.teamtad.com/doku.php/understanding_the_user_interface [Accessed 29 Augustus 2021].
- Duncan, 2020. *BlenderBIM Add-on Clash detection*. [Online] Available at: https://wiki.osarch.org/index.php?title=BlenderBIM_Add-on_Clash_detection [Accessed 05 September 2021].
- Duncan & Filipec, J., 2021. *BlenderBIM Add-on for building and exporting an IFC model*. [Online] Available at: https://wiki.osarch.org/index.php?title=BlenderBIM_Add-on_for_building_and_exporting_an_IFC_model [Accessed 06 September 2021].

Flynn, K., 2018. *Open Source for Everyone?* [Online] Available at: www.architectmagazine.com/aia-architect/aiafuture/open-source-for-everyone_o [Accessed 07 August 2021].

FOSSA, I., 2021. *Open Source Software Licenses 101: The AGPL License*. [Online] Available at: <https://fossa.com/blog/open-source-software-licenses-101-agpl-license/> [Accessed 16 October 2021].

Foster, E., 2020. *How to Write a Good Documentation: Home*. [Online] Available at: <https://guides.lib.berkeley.edu/how-to-write-good-documentation> [Accessed 05 August 2021].

Francis, S., 2017. *Settings*. [Online] Available at: http://docs.teamtad.com/lib/exe/detail.php/settings_system_tab.jpg?id=settings [Accessed 02 September 2021].

Francis, S., 2017. *The Benefits of TAD*. [Online] Available at: http://docs.teamtad.com/doku.php/the_benefits_of_tad [Accessed 05 September 2021].

Francis, S., 2020. *About TAD Designer*. [Online] Available at: http://docs.teamtad.com/doku.php/about_tad_designer [Accessed 04 September 2021].

Francis, S., 2020. *Concepts inside TAD*. [Online] Available at: http://docs.teamtad.com/doku.php/tad_concepts [Accessed 05 September 2021].

Francis, S., 2020. *Open sourcing TAD*. [Online] Available at: <https://community.osarch.org/discussion/243/open-sourcing-tad> [Accessed 05 September 2021].

Francis, S., 2021. *Collaboration*. [Online] Available at: <http://docs.teamtad.com/doku.php/collaboration> [Accessed 05 September 2021].

Francis, S., 2021. *Exporting objects to 3D*. [Online] Available at: [http://docs.teamtad.com/doku.php/exporting_objects_to_3d?s\[\]=material](http://docs.teamtad.com/doku.php/exporting_objects_to_3d?s[]=material) [Accessed 04 September 2021].

Francis, S., 2021. *TAD DESIGNER LITE END USER LICENSE AGREEMENT and TERMS OF USE*. [Online] Available at: <http://docs.teamtad.com/doku.php/tos> [Accessed 03 September 2021].

FreeCAD, 2019. *Raytracing templates*. [Online] Available at: https://wiki.freecadweb.org/Raytracing_templates [Accessed 04 September 2021].

FreeCAD, 2020. *Addon*. [Online] Available at: <https://wiki.freecadweb.org/Addon> [Accessed 01 September 2021].

FreeCAD, 2020. *BIM Library*. [Online] Available at: https://wiki.freecadweb.org/BIM_Library [Accessed 06 September 2021].

FreeCAD, 2021. *About FreeCAD*. [Online] Available at: https://wiki.freecadweb.org/About_FreeCAD [Accessed 01 September 2021].

FreeCAD, 2021. *Arch Equipment*. [Online] Available at: https://wiki.freecadweb.org/Arch_Equipment [Accessed 02 September 2021].

FreeCAD, 2021. *Arch IFC*. [Online] Available at: https://wiki.freecadweb.org/Arch_IFC [Accessed 05 September 2021].

FreeCAD, 2021. *Arch MultiMaterial*. [Online] Available at: https://wiki.freecadweb.org/Arch_MultiMaterial [Accessed 03 September 2021].

FreeCAD, 2021. *Arch Schedule*. [Online] Available at: https://wiki.freecadweb.org/Arch_Schedule [Accessed 04 September 2021].

FreeCAD, 2021. *Arch SetMaterial*. [Online] Available at: https://wiki.freecadweb.org/Arch_SetMaterial [Accessed 03 September 2021].

FreeCAD, 2021. *Arch Survey*. [Online] Available at: https://wiki.freecadweb.org/Arch_Survey [Accessed 05 September 2021].

FreeCAD, 2021. *Arch Workbench*. [Online] Available at: https://wiki.freecadweb.org/Arch_Workbench [Accessed 04 September 2021].

FreeCAD, 2021. *BIM Workbench*. [Online] Available at: https://wiki.freecadweb.org/BIM_Workbench [Accessed 05 September 2021].

FreeCAD, 2021. *Draft Workbench*. [Online] Available at: https://wiki.freecadweb.org/Draft_Workbench [Accessed 06 September 2021].

FreeCAD, 2021. *External workbenches*. [Online] Available at: https://wiki.freecadweb.org/External_workbenches [Accessed 02 September 2021].

FreeCAD, 2021. *History*. [Online] Available at: <https://wiki.freecadweb.org/History> [Accessed 02 September 2021].

FreeCAD, 2021. *History*. [Online] Available at: <https://wiki.freecadweb.org/History> [Accessed 05 September 2021].

FreeCAD, 2021. *Import Export*. [Online] Available at: https://wiki.freecadweb.org/Import_Export [Accessed 02 September 2021].

FreeCAD, 2021. *Interface*. [Online] Available at: <https://wiki.freecadweb.org/Interface> [Accessed 22 Augustus 2021].

FreeCAD, 2021. *Interface Customization*. [Online] Available at: https://wiki.freecadweb.org/index.php?title=Interface_Customization&action=history [Accessed 24 Augustus 2021].

FreeCAD, 2021. *Licence*. [Online] Available at: <https://wiki.freecadweb.org/Licence> [Accessed 03 September 2021].

FreeCAD, 2021. *Manual: BIM modeling*. [Online] Available at: https://wiki.freecadweb.org/Manual: BIM_modeling [Accessed 05 September 2021].

FreeCAD, 2021. *Manual: Creating renderings*. [Online] Available at: https://wiki.freecadweb.org/Manual: Creating_renderings [Accessed 05 September 2021].

FreeCAD, 2021. *Manual: The Community*. [Online] Available at: https://wiki.freecadweb.org/Manual: The_Community [Accessed 02 September 2021].

FreeCAD, 2021. *Manual: The FreeCAD Interface*. [Online] Available at: https://wiki.freecadweb.org/Manual: The_FreeCAD_Interface [Accessed 06 September 2021].

FreeCAD, 2021. *Material*. [Online] Available at: <https://wiki.freecadweb.org/Material> [Accessed 03 September 2021].

- FreeCAD, 2021. *Parts Library Workbench*. [Online] Available at: https://wiki.freecadweb.org/Parts_Library_Workbench [Accessed 06 September 2021].
- FreeCAD, 2021. *Raytracing Workbench*. [Online] Available at: https://wiki.freecadweb.org/Raytracing_Workbench [Accessed 03 September 2021].
- FreeCAD, 2021. *Spreadsheet Workbench*. [Online] Available at: https://wiki.freecadweb.org/Spreadsheet_Workbench [Accessed 02 September 2021].
- FreeCAD, 2021. *Tutorials*. [Online] Available at: <https://wiki.freecadweb.org/Tutorials> [Accessed 06 September 2021].
- FreeCAD, 2021. *Workbenches*. [Online] Available at: <https://wiki.freecadweb.org/Workbenches> [Accessed 01 September 2021].
- Goldstein, A., 2021. *Open Source Licenses Explained*. [Online] Available at: <https://www.whitesourcesoftware.com/resources/blog/open-source-licenses-explained/> [Accessed 03 August 2021].
- Gonzalez-Barahona, J., 2021. A Brief History of Free, Open Source Software and Its Communities. *Computer*, pp.75-79.
- Government, U.S., 2019. *The Open Government Partnership*. [Online] Available at: <https://open.usa.gov/assets/files/NAP4-fourth-open-government-national-action-plan.pdf> [Accessed 05 September 2021].
- Greene, T., 2001. *Ballmer:Linux is a cancer*. [Online] Available at: www.theregister.com/2001/06/02/ballmer_linux_is_a_cancer/ [Accessed 21 August 2021].
- Hanganu, G., 2014. *The Community Source Development Model*. [Online] Available at: <http://oss-watch.ac.uk/resources/communitysource> [Accessed 12 August 2021].
- Havre, Y.V., 2021. *FreeCAD Render Workbench*. [Online] Available at: <https://github.com/FreeCAD/FreeCAD-render> [Accessed 05 September 2021].
- He, G., 2011. *Overview of BIM*. Beijing: China Construction Industry Press.
- Heavey, C. & Dagkakis, G., 2015. A review of open source discrete event simulation software for operations research. *Journal of Simulation*.
- Heavey, C. & Dagkakis, G., 2015. A review of open source discrete event simulation software for operations research. *Journal of Simulation*.
- Holmström, B., 1999. *Managerial Incentive Problems: A Dynamic Perspective*. *Review of Economic Studies*.
- Hurt, J., 2019. *LibreCAD and the GPLv2*. [Online] Available at: <https://github.com/LibreCAD/LibreCAD/tree/master/licenses> [Accessed 05 September 2021].
- Initiative, O.S., 2007. *The Open Source Definition*. [Online] Available at: www.opensource.org/osd [Accessed 16 August 2021].
- Kowalczyk, M., 2017. *IFC Export Roofs problem*. [Online] Available at: <https://forums.autodesk.com/t5/revit-architecture-forum/ifc-export-roofs-problem/td-p/6843152> [Accessed 05 September 2021].

Languages, S.I.G.o.P., 2015. *BYLAWS of the Special Interest Group on PROGRAMMING LANGUAGES of the Association for Computing Machinery, Inc.* [Online] Available at: <https://www.acm.org/special-interest-groups/volunteer-resources/bylaws/sigplan-bylaws> [Accessed 03 August 2021].

Leadbeater, C., 2008. *We Think: The Power Of Mass Creativity*. London: Profile Books.

LibreCAD.org, 2018. *Configuration*. [Online] Available at: <https://librecad.readthedocs.io/en/latest/getstart/configure.html> [Accessed 25 Augustus 2021].

LibreCAD.org, 2018. *Main Menu*. [Online] Available at: <https://librecad.readthedocs.io/en/latest/ref/menu.html?highlight=layout#drawings> [Accessed 26 Augustus 2021].

LibreCAD.org, 2019. *License*. [Online] Available at: <https://librecad.readthedocs.io/en/latest/appx/licenses.html> [Accessed 06 September 2021].

LibreCAD.org, 2020. *Blocks*. [Online] Available at: https://wiki.librecad.org/index.php/Part_Libraries [Accessed 05 September 2021].

LibreCAD.org, 2020. *Building from Source*. [Online] Available at: <https://librecad.readthedocs.io/en/latest/appx/build.html> [Accessed 02 September 2021].

LibreCAD.org, 2020. *LibreCAD User Manual*. [Online] Available at: <https://librecad.readthedocs.io/en/latest/> [Accessed 04 September 2021].

LibreCAD.org, 2020. *The Application*. [Online] Available at: <https://librecad.readthedocs.io/en/latest/about/index.html> [Accessed 03 September 2021].

Logothetis, S. & Stylianidis, E., 2016. BIM open source software (OSS) for the documentation of cultural heritage. . *Virtual Archaeology Review*, pp.28-35.

Manual, F., 2021. *Manual: The Community*. [Online] Available at: https://wiki.freecadweb.org/Manual:The_Community [Accessed 06 September 2021].

Mishra, B.K., Prasad, A. & Raghunathan, S., 2002. Quality and Profits Under Open Source Versus Closed Source. In *ICIS 2002 Proceedings*. Barcelona, 2002.

Moult, D., 2020. *BlenderBIM Add-on Adding labels linked to properties and quantities*. [Online] Available at: https://wiki.osarch.org/index.php?title=BlenderBIM_Add-on_Adding_labels_linked_to_properties_and_quantities [Accessed 05 September 2021].

Moult, D., 2020. *BlenderBIM Add-on Setting up a BIM Project*. [Online] Available at: https://wiki.osarch.org/index.php?title=BlenderBIM_Add-on_Setting_up_a_BIM_Project [Accessed 05 September 2021].

Moult, D., 2021. *BlenderBIM Add-on*. [Online] Available at: https://wiki.osarch.org/index.php?title=BlenderBIM_Add-on [Accessed 05 September 2021].

Moult, D., 2021. *BlenderBIM Add-on Roadmap*. [Online] Available at: https://wiki.osarch.org/index.php?title=BlenderBIM_Add-on_Roadmap [Accessed 02 September 2021].

NBIMS, 2007. *United States National Building Information Modeling Standard. Version 1 - Part 1: Overview, Principles, and Methodologies*. National Institute of Building Sciences.

Nicholson, D., 2018. *Open Source*. [Online] Available at: opensource.org [Accessed 01 August 2021].

Opensource.org, 2007. *The Open Source Definition*. [Online] Available at: <https://opensource.org/osd> [Accessed 12 October 2021].

Perens, B., 1998. Open Source Definition. January.

Petrie, R., 2017. *What is OpenBIM?* [Online] Available at: <https://www.buildingsmart.org/about/openbim/openbim-definition/> [Accessed 12 August 2021].

Quirk, V., 2013. *Paperhouses: Architecture in Open Source*. [Online] Available at: <https://www.archdaily.com/452176/paperhouses-architecture-in-open-source> [Accessed 06 August 2021].

Randhawa, S., 2018. Open Source Software and Libraries.

Regoli, N., 2015. *7 Main Advantages and Disadvantages of Open Source Software*. [Online] Available at: <https://connectusfund.org/7-main-advantages-and-disadvantages-of-open-source-software> [Accessed 15 August 2021].

RIBA, 2020. *RIBA plan of work 2020 overview*. London: Published by RIBA, 66 Portland Place, London, W1B 1AD.

Stallman, R., 2015. *FLOSS and FOSS*. [Online] Available at: <https://www.gnu.org/philosophy/floss-and-foss.en.html> [Accessed 17 August 2021].

Stebich, A., 2019. *docs*. [Online] Available at: <https://github.com/LibreCAD/docs> [Accessed 02 September 2021].

Taher, A.A., 2016. *BIM Software Capability and Interoperability Analysis*.

Turnbull, J., 2018. *Documentation as a gateway to open source*. [Online] Available at: <https://increment.com/documentation/documentation-as-a-gateway-to-open-source/> [Accessed 05 August 2021].

LIST OF ACRONYMS AND ABBREVIATIONS

BIM	Building Information Modelling
OSS	Open Source Software
FOSS	Free and Open Source Software
GPL	General Public License, General Public License
BSD	Berkeley Software Distribution
MIT	Massachusetts Institute of Technology
AFL	Academic Free License
GNU	GNU's not Unix
AGPL	Affero General Public License
AEC	Architecture, Engineering and Construction
QTO	Quantity Take Off
OSI	Open Source Initiative
OSD	Open Source Definition
ARPANET	Advanced Research Projects Agency Network
RFCs	Research Fund for Coal and Steel
BASIC	Beginner's All-purpose Symbolic Instruction Code
BBS	Bulletin board system
FSF	Free Software Foundation
HTTP	HyperText Transfer Protocol
LAMP	Linux, Apache, MySQL, PHP
URL	Uniform Resource Locator
JSON	JavaScript Object Notation
LGPL	Lesser General Public License
EPL	Eclipse Public License
MPL	Mozilla Public License
OSArc	Open Source Architecture
CAD	Computer-Aided Design
IFC	Industry Foundation Classes
BCF	BIM Collaboration Format
COBie	Construction Operations Building Information Exchange
TAD	The Architect's Desktop
GOM	Graphical Object Modular

