PROPOSTA DE PLATAFORMA DE ARQUIVOS BIM SEGUROS EM NUVEM

GUSTAVO MADRUGA JORGE

VICTOR ALBÉDIO COSTA LOPES

PROJETO FINAL DE GRADUAÇÃO EM ENGENHARIA DE REDES DE COMUNICAÇÃO

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SOFTWARE PROPOSAL FOR SECURE CLOUD BIM FILES

PROPOSTA DE PLATAFORMA DE ARQUIVOS BIM SEGUROS EM NUVEM

GUSTAVO MADRUGA JORGE

VICTOR ALBÉDIO COSTA LOPES

ORIENTADOR: GEORGES DANIEL AMVAME NZE
COORIENTADOR: CALVIN MARIANO RÊGO CRISPIM

PROJETO FINAL DE GRADUAÇÃO EM
ENGENHARIA DE REDES DE COMUNICAÇÃO

PUBLICAÇÃO: PPGEA.TD-001/11

BRASÍLIA/DF: MAIO - 2021

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

PROPOSTA DE PLATAFORMA DE ARQUIVOS BIM SEGUROS EM NUVEM

GUSTAVO MADRUGA JORGE

VICTOR ALBÉDIO COSTA LOPES

PROJETO FINAL DE GRADUAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE REDES DE COMUNICAÇÃO.

APROVADA POR:

_____

Prof. Dr. Georges Daniel Amvame Nze – ENE/Universidade de Brasília
Orientador

_____

Prof. Dr. Fábio Lúcio Lopes Mendonça – ENE/Universidade de Brasília
Membro Interno

_____

Calvin Mariano Rêgo Crispim – ENE/Universidade de Brasília
Membro Externo

BRASÍLIA, 26 DE MAIO DE 2021.

FICHA CATALOGRÁFICA

JORGE, GUSTAVO MADRUGA; COSTA, VICTOR ALBÉDIO LOPES
Proposta de Plataforma de Arquivos BIM Seguros em Nuvem [Distrito Federal] 2021.
xi, 47p., 210 x 297 mm (ENE/FT/UnB, Engenheiro de Redes de Comunicação, Engenharia de Redes de Comunicação, 2021).
Projeto Final de Graduação – Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. BIM	2. CBCAD
3. FreeCAD	4. Criptografia
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

JORGE, G. M., COSTA, V. A. L. (2021). Proposta de Plataforma de Arquivos BIM Seguros em Nuvem . Projeto Final de Graduação em Engenharia de Redes de Comunicação, Publicação PPGEA.TD-001/11, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 47p.

CESSÃO DE DIREITOS

Gustavo Madruga Jorge

Victor Albédio Costa Lopes

Departamento de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

Dedico este trabalho aos meus pais, meu irmão e a todos que me auxiliaram nessa longa jornada.

Gustavo Madruga Jorge

Dedico aos meus pais, por nunca terem medido esforços para me proporcionar o necesssário, independentemente de qualquer dificuldade.

Victor Albédio Costa Lopes

AGRADECIMENTOS

Agradeço primeiramente a Deus por sempre iluminar e proteger o meu caminho. Agradeço a toda minha família, que sempre me apoiou e me deu todo o suporte necessário durante todos os anos de graduação. Agradeço aos meus amigos de curso, que me auxiliaram e motivaram durante todo este percurso. Agradeço ao Victor por todo companheirismo e respeito durante a elaboração desse projeto. Agradeço ao professor Dr. Georges e ao mestre Calvin por toda orientação, paciência, confiança e pela oportunidade de realizar este projeto.

Gustavo Madruga Jorge

A Deus, pela minha vida, e por me permitir ultrapassar todos os obstáculos encontrados ao longo da realização deste trabalho.
Aos familiares e a minha noiva, por todo o apoio e pela ajuda, que muito contribuíram para a realização deste trabalho.
Ao professor Georges e ao Mestre Calvin, por terem sido nossos orientadores e terem desempenhado tal função com dedicação e amizade.
Aos meus colegas de curso, com quem convivi intensamente durante os últimos anos, pelo companheirismo e pela troca de experiências que me permitiram crescer não só como pessoa, mas também como formando.

Victor Albédio Lopes Costa

RESUMO

Título: Proposta de Plataforma de Arquivos BIM Seguros em Nuvem
Autor: Gustavo Madruga Jorge
Autor: Victor Albédio Costa Lopes
Orientador: Georges Daniel Amvame Nze
Coorientador: Calvin Mariano Rêgo Crispim
Programa de Graduação em Engenharia de Redes de Comunicação
Brasília, 26 de maio de 2021

Hoje em dia, os grandes centros urbanos estão cada vez mais se desenvolvendo. A engenharia e arquitetura devem evoluir ao mesmo passo ou até de forma mais dinâmica para conseguir construir e manter edificações, prédios, centros comerciais e residenciais que abriguem e suportem tamanho contingente populacional. E toda construção que se preze deve ter um seguro e confiável sistema de prevenção de incêndio. Este projeto propõe uma plataforma segura de busca, abertura, manuseio e visualização de arquivos BIM no software FreeCAD. O projeto visa facilitar o trabalho de quem está numa central de alarmes de incêndio, corpo de bombeiros ou qualquer outro órgão responsável que utilize a arquitetura de visualizações de arquivos BIM. Trata-se de uma efetiva colaboração para os sistemas inteligentes de detecção de incêndio, simplificando comandos, automatizando processos e agregando diversas funcionalidades, que otimizam de forma significativa tempo e esforços. O projeto agrega muito dinamismo e praticidade para que nos momentos mais tensos e delicados de combate a incêndio, todo o sistema funcione de forma mais ágil. É utilizada a linguagem de programação Python para automatizar diversos processos que para o usuário comum pode ser complicado de compreender e executar. Haverá um manual de instruções da plataforma, bem como todos os passos e comandos necessários para a instalação e utilização da forma mais correta e simplificada possível.

Palavras-chave: BIM, CBCAD, FreeCAD, Criptografia.

ABSTRACT

Title: Software Proposal for secure Cloud BIM Files
Author: Gustavo Madruga Jorge
Author: Victor Albédio Costa Lopes
Supervisor: Georges Daniel Amvame Nze
Co-Supervisor: Calvin Mariano Rêgo Crispim
Graduate Program in Network Engineering
Brasília, May 26th, 2021

Nowadays, large urban centers are increasingly developing. Engineering and architecture must progress at the same pace or even more dynamically to be able to construct and maintain buildings, commercial and residential centers that sustain such a contingent population size. And every self-respecting construction must have a safe and reliable fire prevention system. This project proposes a secure platform for searching, opening, handling and viewing BIM files in FreeCAD software. The project aims to facilitate the work of those who are in a fire alarm center, fire department or any other responsible institution that uses the architecture of visualization of BIM files. It is an effective collaboration for intelligent fire detection systems, simplifying commands, automating processes and adding several functionalities, which significantly optimize time and efforts. The project adds a lot of dynamism and practicality so that in the most tense and delicate moments of fire fighting, the whole system works in a more agile way. The Python programming language is used to automate various processes that for the average user can be complicated to understand and execute. There will be an instruction manual for the platform, as well as all the steps and commands necessary for installation and use in the most correct and simplified way possible.

Keywords: BIM, CBCAD, FreeCAD, Criptography.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ACRÔNIMOS E ABREVIA CÕES

ABNT Associação Brasileira de Normas Técnicas.
AEC Arquitetura, Engenharia e Construção.
AIA *American Institute of Architects*.
APP *Application*.
AWS *Amazon Web Services*.

BIM *Building Information Modeling*.

CBMDF Corpo de Bombeiros Militar do Distrito Federal.
CSPRNG *Cryptographically Secure Pseudo-Random Number Generator*.

DBMS *DataBase management system*.
DXF *Drawing exchange format*.

IAI *International Alliance for Interoperability*.
IOT *Internet of things*.
IP *Internet Protocol*.
ISO *International Organization for Standardization*.

NBR Norma Brasileira.
NONCE *Number + Once* .

PHP *Hypertext Preprocessor*.

RAM *Random Access Memory*.

SFN *Secure FireNet*.
SHA *Secure Hash Algorithm*.
SQL *Structured Query Language*.

TCP *Transmission Control Protocol*.

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Não há necessidade de justificar a importância de aprimorar os cuidados e prevenção contra incêndios em prédios. O controle sobre o fogo é um desafio que sempre existiu. E quando se perde esse controle, graves acidentes acontecem. Há diversos exemplos no Brasil e no mundo de trágicos acidentes a partir de incêndios, e à medida que esses acidentes acontecem, as normas e cuidados com esse tema se aprimoram.

A partir disso, surge a motivação para a elaboração desse projeto. A intenção é contribuir de maneira efetiva com o Corpo de Bombeiros e/ou qualquer órgão do Estado na proteção de pessoas e patrimônios. A elaboração de um eficaz projeto é o principal caminho para evitar e combater incêndios. Portanto, a elaboração de bons projetos arquitetônicos ajudam a criar rotas de fugas do prédio por saídas de emergência e evitam ou diminuem o crescimento e alastramento de fogo dentro da edificação e para as edificações vizinhas.

Esse projeto serve de apoio para a dissertação de mestrado "Proposta de Arquitetura Segura de Centrais de Incêndio em Nuvem" de Crispim, Calvin, publicada pelo Departamento de Engenharia Elétrica da Universidade de Brasília em dezembro de 2020. Em sua dissertação, Calvin menciona como principais contribuições futuras para sua arquitetura as seguintes ações:

• A implantação e teste do sistema em centros de operações.
• Complemento do FreeCAD para torná-lo mais eficiente na execução de arquivos IFC.
• Implantação de uma aplicação em nuvem para visualização dos projetos BIM.
• Análise do aproveitamento das informações para perícia.
• Testes de vulnerabilidades da arquitetura desenvolvida.
O projeto busca contribuir com cada uma dessas especificações sugeridas, obviamente dando ênfase em alguns aspectos específicos, como tornar o FreeCAD mais eficiente e rápido.

1.2 OBJETIVOS GERAIS

Auxiliar e facilitar de forma prática o usuário comum a administrar e visualizar os arquivos BIM no software FreeCAD, além de oferecer mecanismos seguros de manutenção de

dados e arquivos.

1.3 OBJETIVOS ESPECÍFICOS

- Automatizar a abertura de arquivos IFC no FreeCAD.
- Tornar a visualização de arquivos no FreeCAD mais clara e limpa.
- Facilitar a inserção de dados de usuário e senha no FreeCAD.
- Destacar sensores que eventualmente estejam ativos.
- Criptografar dados de usuários, arquivos e pastas.

1.4 JUSTIFICATIVA

O projeto em análise está focado no sistema de detecção e alarme de incêndio. Para tal, desenvolveu-se todo um sistema de aproveitamento das informações geradas pelas Centrais Endereçáveis de Incêndios em prol das ações operacionais dos bombeiros. As centrais de incêndio são tratadas pelas normas ABNT NBR ISO 7240-1, que tem a ver com os equipamentos e ABNT NBR 17240, que trata da prestação de serviços em caso de incêndio. Essas normas indicam a possibilidade da Central de Incêndio determinar o sensor, o tipo e localização dos mesmos. Desse modo, a proposta busca explorar tais informações através de um sistema seguro de obtenção, empacotamento e envio das mesmas para emprego em um sistema específico e de uso exclusivo de órgãos responsáveis. Apesar de ser explicitamente referenciado o Corpo de Bombeiros Militar do Distrito Federal – CBMDF como exemplo de estrutura corporativa, o objeto deste estudo pode ser aplicado a qualquer organização que porventura implemente a estrutura preconizada.

Para o desenvolvimento de todo esse projeto, uma condição é fundamental: a segurança das informações. Com o avanço constante da tecnologia IoT (*Internet Of Things*) e a evolução computacional, a interação entre os mais diversos dispositivos só aumenta, e com isso, os desafios de se manter a integridade das informações trocadas aumentam no mesmo passo.

Esse progresso computacional também permite o desenvolvimento de técnicas de modelagem, armazenamento e troca de informações de forma segura. Portanto, o que é produzido na central de incêndio pode ser analisado, encapsulado de forma segura, transmitido e mantido em banco de dados de forma a manter a propriedade e integridade de toda a informação.

1.5 ORGANIZAÇÃO

Este trabalho esta dividido em 6 capítulos. Esse primeiro é a introdução e no segundo está a fundamentação teórica do projeto. No terceiro capítulo, descrevemos a implementação e todos os passos para a utilização do software. No quarto capítulo, apresentamos os resultados obtidos nas aplicações. Por fim, no quinto capítulo fazemos uma breve conclusão do que conseguimos alcançar e no sexto citamos todas as referências bibliográficas nas quais nos baseamos.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 BIM

BIM é a abreviação para *Building Information Modeling*, que é uma das metodologias mais promissoras dos últimos tempos na indústria da arquitetura, engenharia e construção. Utilizando essa tecnologia, é possível criar digitalmente um modelo virtual parametrizado preciso de um projeto de construção, aproveitando praticamente todos os elementos e informações do projeto. Dessa forma, se torna uma importante ferramenta para arquitetos, engenheiros e construtores, visualizando o que deve ser construído em um ambiente simulado para identificar qualquer projeto, construção ou problemas operacionais em potencial. O BIM representa um novo paradigma dentro da indústria de arquitetura, engenharia e construção, que incentiva a integração das funções de todas as partes interessadas em um projeto, economizando tempo e evitando erros de comunicação que se traduzem em desperdício e atraso nas obras.[2]

O BIM pode ser visto como um processo virtual que abrange todos os aspectos, disciplinas e sistemas de uma instalação em um único modelo virtual. Em softwares que aplicam esse conceito, vários profissionais podem usar o mesmo arquivo para trabalhar no mesmo projeto ao mesmo tempo, adicionar dados que correspondam à sua especialidade e visualizar atualizações no modelo em tempo real. Conforme o modelo está sendo criado, os membros da equipe estão constantemente refinando e ajustando suas porções de acordo com as especificações do projeto e alterações de design para garantir que o modelo seja o mais preciso possível antes que o projeto seja fisicamente desenvolvido.[2]

O projeto ideal realizado em BIM deve reunir todas as partes relevantes do planejamento arquitetônico, fornecer informações detalhadas sobre cada detalhe da construção e estar disponível para todos os envolvidos, desde engenheiros e arquitetos, até planejadores e pessoas responsáveis pela compra de materiais.[4]

Figura 2.1 – Ciclo BIM

Fonte: Gmartiniengenharia

2.1.1 Funcionalidades BIM

A tecnologia BIM oferece o potencial para atingir objetivos sempre buscados pela indústria de arquitetura, engenharia e construção, como diminuição de custo do projeto, aumento da produtividade e qualidade e redução do tempo de entrega do projeto. O projeto BIM contém geometria precisa e dados relevantes necessários para apoiar as atividades de projeto, aquisição, fabricação e construção necessárias para realizar a construção. Após a conclusão, este modelo pode ser usado para fins de operação e manutenção. Um modelo em BIM indica a geometria, relações espaciais, informações geográficas, quantidades e propriedades de elementos de construção, propriedades térmicas e acústicas, estimativas de custo, estoques de materiais e cronograma do projeto. O modelo pode ser usado para demonstrar todo o ciclo de vida da construção. Como resultado, quantidades e propriedades compartilhadas de materiais podem ser facilmente extraídas. Os escopos de trabalho podem ser facilmente isolados e definidos. Sistemas, montagens e sequências podem ser mostrados em uma escala relativa dentro de toda a instalação ou grupo de instalações. Documentos de construção como desenhos, detalhes de aquisição, processos de envio e outras especificações podem ser facilmente inter-relacionados.[2]

Figura 2.2 – Capacidade BIM

Fonte: Pinterest/ProcessoBIM

É importante observar que BIM não é apenas software; é um processo e software. BIM significa não apenas usar modelos inteligentes tridimensionais, mas também fazer mudanças significativas no fluxo de trabalho e nos processos de entrega do projeto. Tem potencial para promover uma maior eficiência e harmonia entre profissionais que, no passado, se viam como adversários. O BIM contribui fortemente para o conceito de entrega de projeto integrado, que é uma nova abordagem de entrega de projeto para integrar pessoas, sistemas e estruturas. É parte importante do processo colaborativo para reduzir o desperdício e otimizar a eficiência em todas as fases do ciclo de vida do projeto.[2]

2.1.2 Aplicações BIM

Os softwares mais populares nos meios de engenharia e arquitetura, as ferramentas CAD, como o *AutoCAD*, *SketchUp*, *Solidworks*, etc. ficaram marcadas como ferramentas muito importantes na transição da prancheta para o computador, fornecendo plataformas de desenho 2D para arquitetos e engenheiros. Essas ferramentas não estagnaram no tempo e estão se adaptando ao modelo BIM. A aplicação desses softwares do segmento mudou, os principais

já devem atuar em BIM, como *Autodesk*, *Revit*, *Freecad* e *ArchiCad*, por exemplo. Já os programas que só atuam no modelo CAD não vão sumir, mas terão um papel secundário para agregar todas as funcionalidades do BIM, já que essa tem o propósito de agregar as diversas partes do projeto.

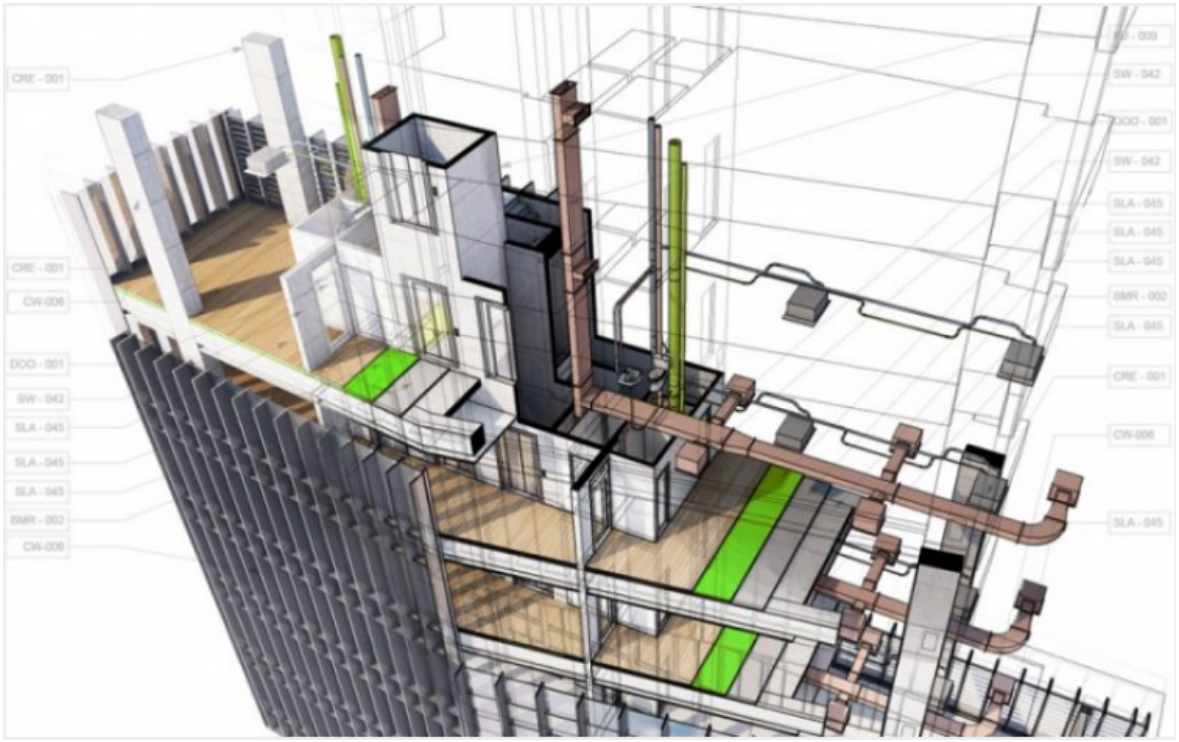

Figura 2.3 – Visualização detalhada de um arquivo BIM

Fonte: Site Renzeti

Se considerarmos que a tecnologia de realidade aumentada está avançando rapidamente e a associarmos ao BIM, em breve, com o uso de smartphones, tablets e ferramentas como o *Google Glass*, poderemos visualizar o projeto no local em que ele será construído antes mesmo de terminá-lo e interagir com ele, entre outras facilidades.[4]

O 3D foi a evolução natural do desenho de projeto, e a BIM potencializa esse avanço. Um objeto modelado em 3D muitas vezes era apenas uma representação do modelo real, uma simulação, sendo apenas uma ferramenta de visualização. Com a aplicação da tecnologia BIM, esse modelo 3D passa a ter diversos objetos paramétricos com a adição das informações que o BIM consegue proporcionar. Ou seja, cada objeto modelado passa a aceitar parâmetros e informações que agregam ao trabalho de outros profissionais que não sejam apenas os projetistas.[4]

Costuma-se dizer no segmento de construção civil que quanto mais próximo da realidade o planejamento da obra estiver, maiores são as chances de sucesso do empreendimento no futuro. As tecnologias que utilizam BIM estão sintonizadas com essa realidade e criaram

recursos para que a visualização prévia das edificações fique ainda mais apurada,detalhada e visível, permitindo planejamentos ainda mais precisos.[4]

2.1.3 Benefícios BIM

O BIM pode ser utilizado de diversas maneiras em vários contextos, algumas de suas inúmeras vantagens são: [2] •Visualização: renderizações 3D podem ser facilmente geradas internamente com pouco esforço adicional.

•Desenhos de fabricação / fábrica: É fácil gerar desenhos de fábrica para vários sistemas de construção. Por exemplo, os desenhos da oficina de dutos de chapa metálica podem ser produzidos rapidamente assim que o modelo estiver completo.

•Revisão do código: Corpo de bombeiros e outros oficiais podem usar esses modelos para a revisão de projetos de construção e após construídos, usar para monitoramento e manutenção.

•Estimativa de custos: o software BIM possui recursos integrados de estimativa de custos. As quantidades de materiais são extraídas e atualizadas automaticamente quando quaisquer alterações são feitas no modelo.

•Sequenciamento de construção: um modelo de informações de construção pode ser usado com eficácia para coordenar pedidos de materiais, fabricação e cronogramas de entrega de todos os componentes de construção.

•Detecção de conflito, interferência e colisão: Como os modelos de informações de construção são criados para escalar no espaço 3D, todos os principais sistemas podem ser verificados de forma instantânea e automática quanto a interferências. Por exemplo, esse processo pode verificar se a tubulação não faz interseção com vigas, dutos ou paredes de aço.

•Análise forense: um modelo de informações de construção pode ser facilmente adaptado para ilustrar graficamente possíveis falhas, vazamentos, planos de evacuação e assim por diante.

•Gerenciamento de instalações: os departamentos de gerenciamento de instalações podem usá-lo para reformas, planejamento de espaço e operações de manutenção. O principal benefício de um modelo de informações de construção é sua representação geométrica precisa das partes de uma construção em um ambiente de dados integrado.

•Processos mais rápidos e eficazes: as informações são mais facilmente compartilhadas e podem ser agregadas com valor e reutilizadas.

•Melhor design: as propostas de construção podem ser rigorosamente analisadas, as simulações realizadas rapidamente e o desempenho aferido, permitindo soluções aprimoradas e inovadoras.

•Custos controlados de toda a vida e dados ambientais: o desempenho ambiental é mais previsível e os custos do ciclo de vida são mais bem compreendidos.

•Melhor qualidade de produção: a produção de documentação é flexível e explora a automação.

•Montagem automatizada: os dados digitais do produto podem ser explorados em processos posteriores e usados para fabricação e montagem de sistemas estruturais.

•Melhor atendimento ao cliente: as propostas são mais bem compreendidas por meio de uma visualização precisa.

•Dados do ciclo de vida: Requisitos, projeto, construção e informações operacionais podem ser usados no gerenciamento de instalações.

2.1.4 Riscos e Desafios

Os desafios e riscos enfrentados pela tecnologia BIM podem ser divididos em duas grandes categorias: legais (ou contratuais) e técnicos.

Assim como na grande maioria das novas tecnologias que se desenvolvem no mundo atual, provavelmente o principal desafio da tecnologia BIM é a segurança e privacidade dos dados. Existe a necessidade de proteger e preservar alguns dados contidos em um projeto por meio de leis de direitos autorais e outros canais legais. Por exemplo, se o proprietário está pagando pelo projeto, ele pode se sentir no direito de possuir todas as informações, mas se os membros da equipe estiverem necessitando e fornecendo informações proprietárias para uso no projeto, suas informações proprietárias também precisam ser protegidas.

O principal desafio nesse caso é definir o que pode ou não ser visto e por quem pode ser visto e editado. Portanto, essa não é uma questão de simples resolução e não há uma resposta ou uma única solução para esse problema. Essa é uma questão singular de cada projeto, dependendo das necessidades dos participantes, e deve ser encarada de forma individualizada. O objetivo é evitar inibições ou desincentivos que desencorajam ou desmotivem os participantes de atingirem o potencial máximo de suas capacidades. Para buscar ao máximo um consenso sobre questões de direitos autorais e privacidade de dados, a melhor solução é estabelecer e definir com clareza nos documentos contratuais os direitos e responsabilidades de propriedade de cada participante do projeto.[10]

Outro ponto desafiador nesse sentido de segurança de dados é saber e decidir qual é a pessoa ou o tipo de profissional mais adequado para controlar a entrada de dados no modelo e será responsável por quaisquer erros, invasões e imprecisões. Assumir a responsabilidade de atualizar os dados do modelo de informações de construção e garantir sua precisão envolve um grande risco. Pedidos de indenizações complicadas por usuários BIM e a oferta de garantias limitadas e isenções de responsabilidade por designers são pontos de negociação

essenciais que precisam ser resolvidos antes que a tecnologia BIM seja implementada ao projeto. [10]

A tecnologia BIM requer mais tempo para inserir e revisar os dados, o que é um novo custo no processo de design e administração do projeto. Embora esses novos custos possam ser drasticamente compensados por ganhos de eficiência e cronograma, eles ainda são um custo adicional que alguém da equipe do projeto terá de arcar. Assim, antes que a tecnologia BIM possa ser totalmente utilizada, não apenas os riscos de seu uso devem ser identificados e alocados, mas também o custo de sua implementação.[11]

Um exemplo de um possível divergência dentro de um projeto BIM pode ocorrer quando algum profissional que não seja o proprietário e o arquiteto/engenheiro contribui com dados integrados ao projeto. Podem surgir problemas de licenciamento. Por exemplo, os fornecedores de equipamentos e materiais oferecem designs associados aos seus produtos para a conveniência do designer principal na esperança de induzir o designer a especificar o equipamento do fornecedor. Embora essa prática possa ser boa para os negócios, podem surgir problemas de licenciamento se os designs não forem produzidos por um designer licenciado no local do projeto.[11]

É evidente que tamanha integração contida no conceito de BIM pode acarretar diversos problemas e se tornar uma bola de neve, acumulando e propagando eventuais erros. Muitas vezes as responsabilidades e compromissos se alteram ou são confundidos. Uma demonstração disso pode ser um cenário no qual o proprietário dos arquivos de construção se ajusta a um erro de projeto percebido. O arquiteto, os engenheiros e outros colaboradores do processo BIM olham uns para os outros em um esforço para tentar determinar quem é o responsável pelo assunto levantado. Se houver desacordo, o profissional líder não apenas será responsável por uma questão de lei para com o reclamante, mas pode ter dificuldade em provar a falha de outros, como os engenheiros.[11]

À medida que as dimensões de custo e cronograma são incorporadas ao modelo de informações de construção, a responsabilidade pela interface tecnológica adequada entre os vários programas torna-se um problema. Muitas equipes sofisticadas de contratação exigem que os subcontratados apresentem cronogramas detalhados do método do caminho crítico e análises de custos discriminadas por itens de linha de trabalho antes do início do projeto. O empreiteiro geral então compila os dados, criando um cronograma mestre e detalhamento de custos para todo o projeto. Quando os subcontratados e o contratante principal usam o mesmo software, a integração pode ser fluida. Nos casos em que os dados estão incompletos ou são enviados em uma variedade de softwares de programação e custo, um membro da equipe - geralmente um empreiteiro geral ou gerente de construção - deve reinserir e atualizar um cronograma mestre e programa de custos. Esse programa pode ser um módulo BIM ou outro programa integrado ao BIM. Atualmente, a maioria dessas ferramentas de gerenciamento de projetos foi desenvolvida isoladamente. A responsabilidade pela precisão e

coordenação dos dados de custo e programação deve ser tratada contratualmente.[10]

Uma das maneiras mais eficazes de lidar com esses riscos é ter contratos de entrega de projeto colaborativos e integrados, nos quais os riscos do uso do BIM são compartilhados entre os participantes do projeto junto com as recompensas. Recentemente, o *American Institute of Architects* lançou uma exposição sobre BIM para ajudar os participantes do projeto a definir seu plano de desenvolvimento de BIM para entrega de projeto integrado. Esta exposição pode ajudar os participantes do projeto na definição de arranjos de gerenciamento de modelo, bem como autoria, propriedade e requisitos de nível de desenvolvimento, em várias fases do projeto.[2]

2.2 IFC

Muitas funcionalidades no mundo real e virtual têm se tornado cada vez mais padronizadas. Alguns esforços de padronização foram muito bem-sucedidos. Na maior parte do mundo, pessoas podem enviar e-mails em qualquer lugar e para qualquer lugar. A maioria dos telefones do mundo agora interopera globalmente. Os usuários da Internet em qualquer lugar do mundo podem recuperar páginas da web de qualquer lugar do mundo. Na prática médica existem padrões de conteúdo conforme os órgãos oficiais especificam “Protocolos” que prescrevem detalhadamente os procedimentos de como tratar diferentes situações médicas. Os padrões tendem a ser de dois tipos: padrões de interface e padrões de conteúdo. Um exemplo do primeiro é um padrão de conexão de dispositivo de computador, como RS232 ou o protocolo TCP / IP. Os protocolos de interface são comuns para dispositivos físicos (por exemplo, largura da linha de trem), sistemas elétricos (por exemplo, RS232) e software (por exemplo, TCP / IP). Alguns padrões de interface emergem do domínio de mercado de uma empresa de produto (por exemplo, formato DXF para arquivos CAD), e alguns são desenvolvidos por organismos de normalização internacionais, como a ISO. O Código de Construção Uniforme é um sucesso exemplo de um padrão de conteúdo. Em comparação com os padrões de interface, os padrões de conteúdo são muito maiores e mais complicados. Eles se referem explicitamente a um vocabulário padrão de entidades conceituais e processos definidos. Nesse contexto de padronização, surge o "*Industry Foundation Class*".[3]

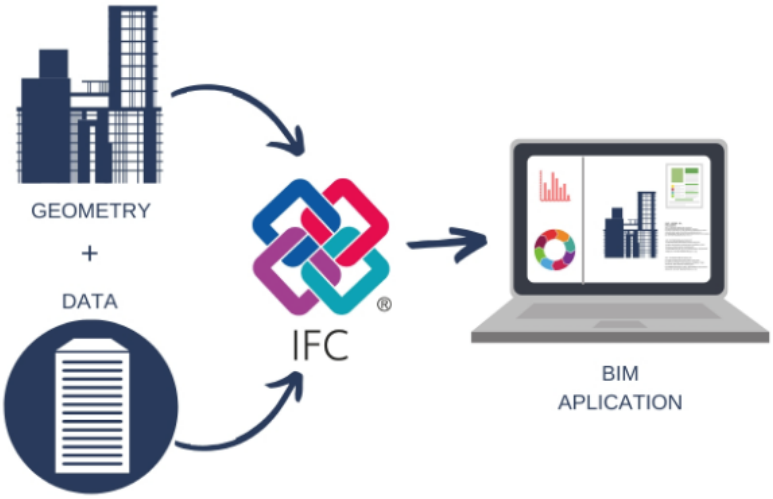

Figura 2.4 – Arquivo IFC

Fonte: Site Bimcorner

Embora um padrão de interface seja muito mais fácil de criar e gerenciar, os IFCs necessariamente têm o sabor de um padrão de conteúdo. Modelos de design muitas vezes são criados inicialmente em software CAD como modelos gráficos de um desenho esquemático ou físico. Designers adicionam anexos semânticos para descrever, relacionar e elaborar entidades gráficas. Antigamente, os aplicativos de software AEC não interoperavam entre si. Um engenheiro teria de revisar manualmente e conferir cada dado da saída de um aplicativo e adaptar e formular a entrada para um aplicativo relacionado. Essa implementação e tradução de dados manuais tomam muito tempo e esforço e a motivação por trás dos IFCs surge a partir dessa questão.[3]

2.2.1 Evolução do IFC

O "*Industry Foundation Class*"é um formato aberto e neutro de dados para o open BIM. IFC foi criado pela *Industry Alliance for Interoperability*(IAI), um consórcio de indústrias de construção e gerenciamento de instalações criado pela Autodesk em 1995, que mais tarde viria a se chamar *BuildingSMART*. A missão da *BuildingSMART* é contribuir para um ambiente sustentável construído por meio de informações mais inteligentes de compartilhamento e comunicação. Isso facilita a coordenação de informações entre aplicativos incompatíveis, que é um pré-requisito para melhorar os fluxos de trabalho de construção usando BIM. O processo elimina o alto custo e resíduos criados por interoperabilidade inadequada.[36]

Em sua essência, *BuildingSMART* permite que toda a indústria de ativos construídos melhore o compartilhamento de informações ao longo do ciclo de vida do projeto ou ativo. A *BuildingSMART* contribiu para o desenvolvimento da maioria das normas internacionais

relacionadas ao BIM, sendo algumas delas: [13]

• ISO 16739:2008: esta norma diz respeito à normalização do IFC.

• ISO 12006-3: 2007: esta norma se refere a uma estrutura de informações orientada a objetos para apoiar a interoperabilidade na indústria da construção.

• ISO 29481-1: 2016 (substitui a ISO 29481-1:2010) e ISO 29481-2: 2012: essas normas especificam um método para coletar e especificar processos e fluxos de informações ao longo do ciclo de vida da instalação.

A primeira sinalização de surgimento do IFC ocorre em 1997, com o IFC 1.0. Após isso, o formato passa por um constante desenvolvimento.

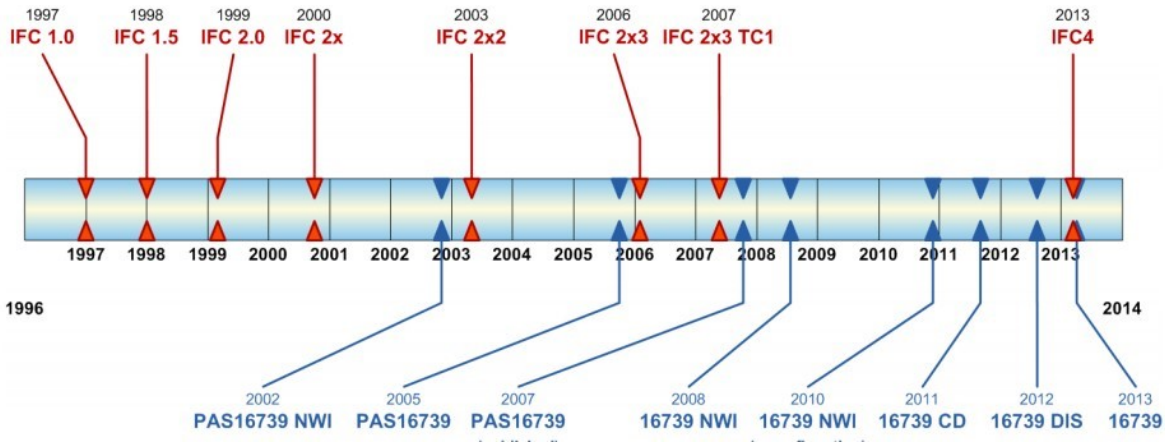

Figura 2.5 – Evolução do formato IFC

Fonte: Site Neo Ipsum

Desde a IFC 1.0, o formato IFC usa a linguagem *EXPRESS* para especificação de dados, padronizada desde 1994 (ISO 10303-11: 1994) e revisada em 2004 (ISO 10303-11: 2004). É uma linguagem usada para dados do produto e pode ser apresentada em dois formatos, texto (ASCII) ou gráfico (*EXPRESS-G*), sendo este último muito mais intuitivo e compreensível.[13]

2.2.2 Formato IFC

O formato IFC foi criado combinando uma série de esquemas de dados na linguagem de modelagem de dados *EXPRESS* em um único esquema interpretável por computador.

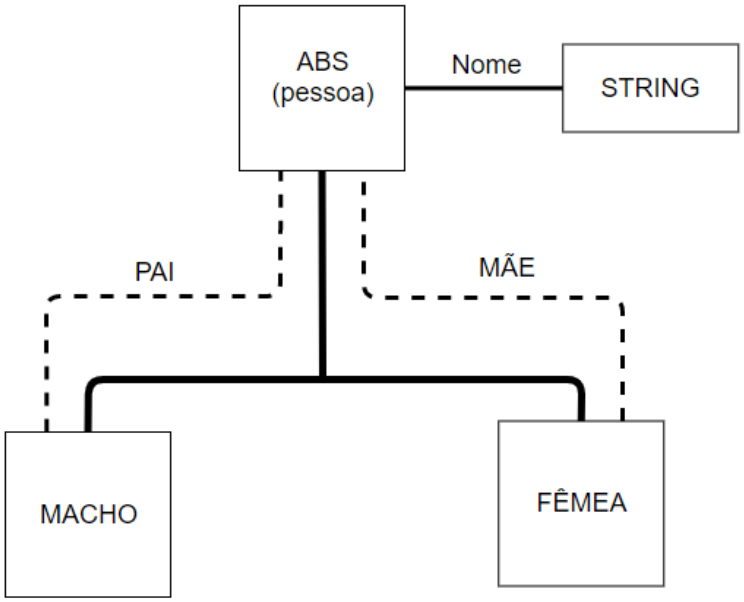

Figura 2.6 – Diagrama de dados da linguagem *EXPRESS-G*.

Fonte: Site Neo Ipsum

Na figura 2.7, podemos observar o esquema de dados específicos do IFC, separados nas seguintes camadas: [13]

• Camada de recursos: a camada inferior, incluindo todas as definições presentes no esquema de dados, e será usada pelas classes na camada superior.

• Camada central (principal): uma camada que inclui um esquema de *kernel* e uma extensão central. Embora o *kernel* forneça conceitos gerais sobre objetos (*IfcObject*), relacionamentos (*IfcRelationship*) e definições (*IfcPropertyDefinition*), a extensão central contém especificações sobre as classes do kernel (*IfcProcess*, *IfcControl* e *IfcProduct*).

• Camada de interoperabilidade: uma camada que inclui módulos que permitem a interoperabilidade e a troca de informações entre diferentes domínios da IFC.

• Camada de domínio: o nível superior contém módulos para vários domínios de construção (por exemplo, arquitetura, engenharia estrutural, simulação de energia), com definições específicas para cada produto, processo ou recurso.

As propriedades mais importantes do formato IFC incluem *IfcPropertySet* e *IfcObject*. Esta classe permite definir as propriedades de um objeto e distingui-las com o prefixo “Pset”, enquanto *IfcObject* define o tipo de objeto físico (por exemplo, parede, janela, piso etc.). [13]

Ao modelar um objeto com base no formato IFC, é possível definir o tipo de objeto a ser modelado e seus atributos. Por exemplo, se um objeto é classificado como IfcBeam, significa que o objeto é um feixe de luz. [13]

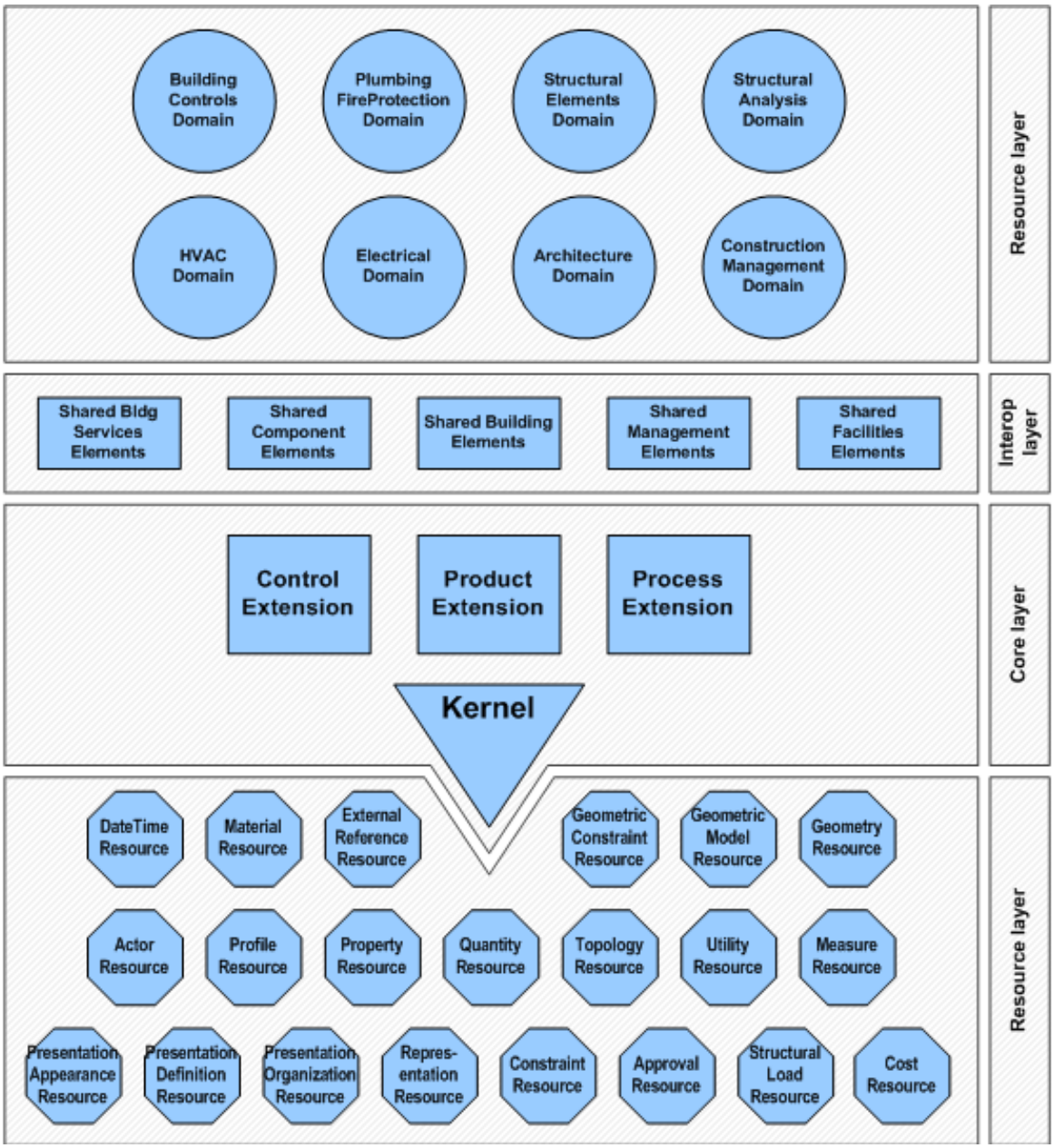

Figura 2.7 – Esquema de dados específicos do IFC, seprados por camadas

Fonte: Site Neo Ipsum

2.2.3 Benefícios do IFC

IFC é uma descrição digital padronizada da indústria de ativos construídos, que promove recursos utilizáveis em uma ampla gama de dispositivos de hardware, plataformas de software e interfaces para muitos casos de uso diferentes. Permite troca de informações entre as diversas aplicações BIM, inclusive de fabricantes diferentes de software. O arquivo IFC compila as informações e propriedades geométricas e não-geométricas de objetos de construção "inteligentes"e captura seus relacionamentos na construção de modelos de informação. O *OpenBIM* é a capacidade de trabalhar em equipe com soluções de software de fabricantes

diferentes. Isso é necessário pois uma solução única de software dificilmente atenderá a todo o projeto que qualquer software “IFC compatível” pode visualizar.

Ao quebrar os silos de informações, os usuários finais podem colaborar e cooperar melhor, independentemente do aplicativo de software que estejam usando. Muito do foco dos IFCs tem sido a representação das instalações que estão sendo projetadas e construídas, mas o escopo dos IFCs também inclui informações de gerenciamento de projetos, como custos, cronogramas, tarefas de trabalho, recursos, etc.

O esquema IFC é um modelo de dados padronizado que codifica, de forma lógica a identidade e semântica (nome, identificador único legível por máquina, tipo de objeto ou função), as características ou atributos (como material, cor e propriedades térmicas), e relacionamentos (incluindo locais, conexões e propriedade), de objetos (como colunas ou lajes), conceitos abstratos (desempenho, custo), processos (instalação, operações), e pessoas (proprietários, designers, empreiteiros, fornecedores, etc.). A especificação do esquema pode descrever como um recurso ou instalação é usado, como é construído e como é operado. A IFC pode definir componentes físicos de edifícios, produtos manufaturados, sistemas mecânicos / elétricos, bem como modelos de análise estrutural mais abstratos, modelos de análise de energia, detalhamento de custos, cronogramas de trabalho e muito mais.[37]

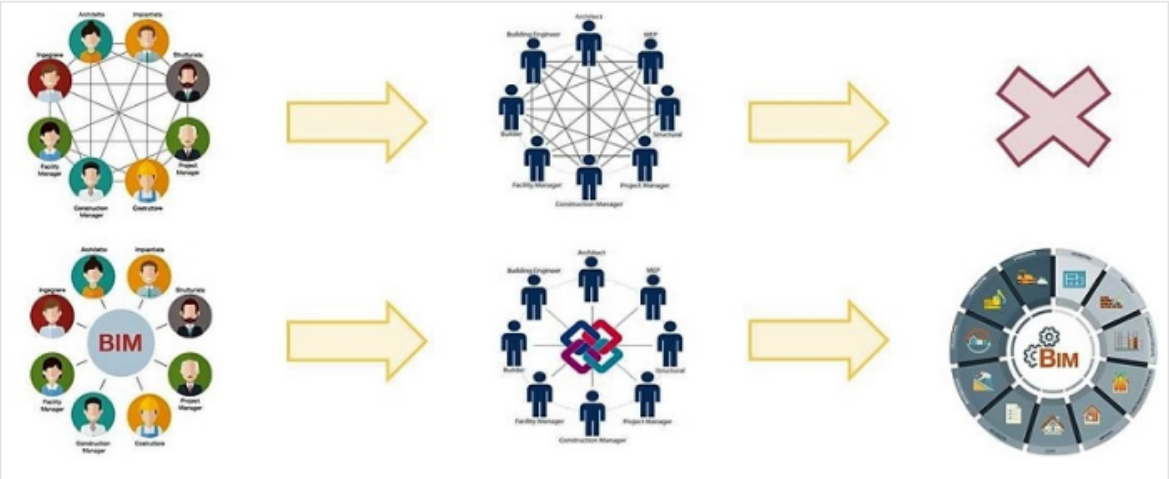

Figura 2.8 – Comparação entre um sistema que não utiliza BIM e IFC e um sistema que utiliza.

Fonte: Site Biblus.accasoftware

2.3 FREECAD

O software de código aberto para equipamentos científicos precisa fornecer arquivos de origem e documentação suficiente para permitir o estudo, replicação e modificação do projeto, e é uma filosofia de criação de aplicativos voltada para a colaboração entre desenvolve-

dores. É aquele cujo design é disponibilizado publicamente para que qualquer pessoa possa estudar, modificar, distribuir, fabricar e vender o design ou software com base nesse design.

Além disso, a modelagem paramétrica é incentivada a fim de facilitar a personalização para outros experimentos. O projeto paramétrico usando uma linguagem de programação de modelagem de sólidos permite a personalização e fornece um arquivo de origem para o projeto. O software de código aberto não só permite um equipamento de laboratório mais acessível, mas também contribui para o desenvolvimento do *Open Science* ao facilitar a replicação, compartilhamento, comparação e até eventuais correções dos experimentos científicos. Além disso, favorece o aprimoramento de experimentos ao permitir que outros editem e desenvolvam dispositivos para diferentes fins.

Alguns benefícios de usar uma ferramenta CAD baseada em script, como o FreeCAD, são permitir a personalização do modelo por meio de um projeto paramétrico e fornecer um código-fonte para o projeto mecânico, tornando-o mais semelhante ao software. Além disso, permite que projetos utilizem ferramentas de gerenciamento de software como versão de controle, documentação de software e ferramentas colaborativas.

Pode-se ter a ideia de que não há vantagem ou qualquer incentivo ao se criar um software código aberto, já que parece não haver muito incentivo, pois teoricamente não há retorno financeiro com as contribuições. Baseado nisso, surge o questionamento acerca do motivo que levaria milhares de programadores de primeira linha a contribuir livremente para o fornecimento de um bem público. Pode sim haver interesse e incentivos privados para pessoas e empresas inovarem e colaborarem. Por exemplo, os empresários usam seus próprios fundos para desenvolver conhecimento e produtos que geram fluxos de receita (e funcionários são pagos por seus serviços criativos à empresa). A sociedade também incentiva a inovação, implementando mecanismos para proteger a propriedade intelectual associada aos produtos, para que fluxos de receita futuros podem ser garantidos para o inovador. Além desse possível interesse privado, existe também o fato dos desenvolvedores poderem ser agraciados com benefícios pessoais com isso, como aprender a desenvolver software complexo, aperfeiçoar a experiência com uma linguagem de computador, melhorar sua reputação, ou por pura diversão e prazer. Muitos desses benefícios dependem da associação a uma comunidade de desenvolvedores que funcione bem. Normalmente, em comunidades de código aberto, membros dão feedback direto, específico e imediato no código de software que outros escrevem e enviam. Esses processos de análise, discussão e revisão pelas partes é não só valioso para o indivíduo que submete o código, mas também para garantir a qualidade geral do software. [16]

2.3.1 Benefícios do FreeCAD

Dentre os diversos softwares existentes na tecnologia BIM, o escolhido para o projeto foi o FreeCAD. Outros softwares analisados se limitavam apenas para o uso nas funcionalidades existentes. O FreeCAD, além de ser gratuito, permite ao programador desenvolver novas funcionalidades ou aprimorar alguma já existente e potencializar a capacidade gráfica de modelos BIM. A comunidade envolvida nessa área é muito colaborativa e os membros costumam compartilhar experiências, dúvidas, soluções, opiniões e diversas informações muito úteis para os desenvolvedores em fóruns abertos. O FreeCAD está em desenvolvimento desde 2002, foi criado para projetos voltados para a engenharia mecânica, mas com o engajamento de toda essa comunidade de desenvolvedores, o software se aplica a diversas áreas da engenharia. Com base em experiências e estudos de softwares, percebemos que o FreeCAD apresenta diversos benefícios quando comparamos com outros, como a sua capacidade de exportar para formatos CAD paramétricos padrão, com isso as informações dimensionais paramétricas do modelo não são perdidas, o uso de uma linguagem de programação difundida com uma extensa biblioteca padrão, como o python e a capacidade de usar e integrar os modelos gerados e os scripts na interface gráfica.[30]

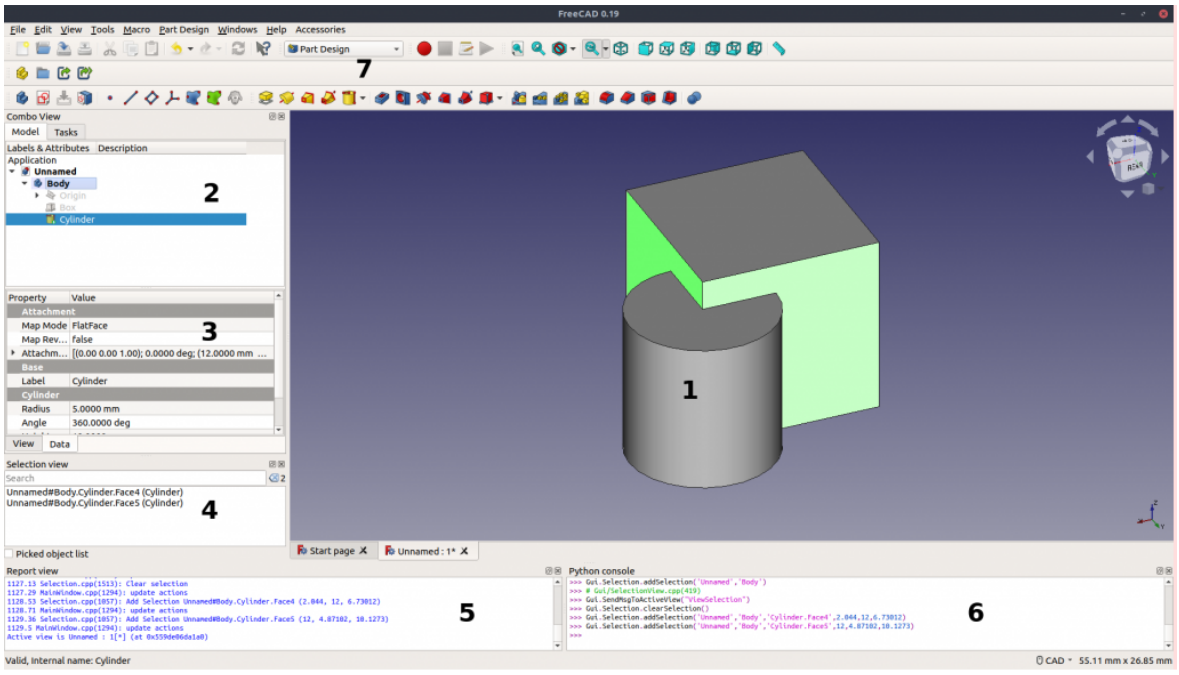

Figura 2.9 – Visualização de um arquivo no FreeCad

Fonte: Site FreeCadweb

2.4 BANCO DE DADOS

Um banco de dados pode ser definido como um conjunto de informações organizadas e geralmente armazenadas eletronicamente em um sistema de computador, e controlado por um sistema de gerenciamento de banco de dados (DBMS). Um DBMS funciona como uma interface entre o banco de dados e seus usuários finais ou programas, dando liberdade ao usuário final, melhorando a visualização dos dados, permitindo que os usuários recuperem, atualizem e gerenciem como as informações são estruturadas, otimizadas e consultadas. Um DBMS também facilita o acesso, domínio e o acompanhamento de bancos de dados, permitindo diversas operações administrativas, como monitoramento de desempenho, ajuste e backup e recuperação. Os dados são coletados e guardados de forma a facilitar ao máximo gerenciamento, modificação, atualização e organização, bem como o controle e segurança das informações, trazendo assim agilidade e eficiência ao sistema. A maioria dos bancos de dados usa a linguagem de consulta estruturada (SQL) para escrever e consultar dados.[26]

2.5 *MYSQL*

O sistema gerenciador de banco de dados escolhido para o projeto foi o *MySQL*. O *MySQL* é um sistema relacional de código aberto que hoje pertence à *Oracle*, e é um dos sistemas mais utilizados em todo o mundo, devido à facilidade no manuseio, estabilidade, ótimo desempenho, portabilidade, compatibilidade, etc. Para utilizar o Python com *MySQL*, utilizamos a biblioteca *PyMySQL*. A biblioteca *PyMySQL* é uma interface cliente *MySQL* escrita em Python puro, baseada na PEP 249 (*Python Enhancement Proposals*), que permite realizar conexão e interação com bancos de dados *MySQL* e *MariaDB*, e implementa o *Python Database API 2.0*, que é e API que especifica o acesso a bancos de dados usando a linguagem Python.[26]

2.6 SEGURANÇA E CRIPTOGRAFIA DE DADOS

Como já mencionado, a segurança das informações é parte primordial desse projeto como um todo. Como em qualquer sistema inteligente hoje em dia que envolva IoT, a segurança dos dados deve ser sempre preservada e aprimorada. Nossa contribuição para esse projeto nessa parte de segurança dos dados foi realizar a criptografia de nomes de usuários, senhas, arquivos BIM e pastas com os arquivos BIM.

As técnicas de criptografia existem desde a antiguidade, quando as mensagens viajavam longas distâncias, tinham apenas um destinatário e só esse destinatário deveria saber e reco-

nhecer a mensagem, e ela não poderia cair em mãos erradas. Dessa forma, quem enviava uma informação era capaz de mesclar as palavras, a fim de que somente quem tinha conhecimento do verdadeiro recado, no caso o receptor, pudesse "desembaralhar" as palavras e ter acesso ao seu conteúdo original.

Nos tempos atuais, a criptografia de dados funciona de forma análoga. São técnicas que transformam informação inteligível em algo que um agente externo seja incapaz de compreender. Desenvolvem-se métodos pensados para proteger uma informação de modo que apenas emissor e receptor consigam compreendê-la, ou um conjunto de pessoas previamente autorizadas, sem que haja acessos indevidos no caminho.

Normalmente, são utilizados protocolos de segurança e chaves, capazes de codificar ou criptografar uma mensagem no momento do envio, e decodificá-la ou descriptografá-la quando chega ao destinatário correto. A criptografia normalmente é associada a assuntos ligados a guerra, mas os governantes também podem ver na criptografia um ótimo meio para evitar que pessoas não autorizadas descubram informações sobre suas estratégias, bem como comerciantes, negociantes, empresários, etc. A criptografia é um método de proteção e privacidade de dados muito importante e confiável e cada vez mais presente.

Com a criptografia, caso alguma mensagem, senha, arquivo ou qualquer forma de informação seja interceptada de forma indevida, seria necessário o conhecimento da chave correta para reconhecer o conteúdo. No caso do interceptor não possuir a autorização para ver o conteúdo, ou seja, não ter acesso à determinada chave, verá apenas uma lista desordenada e aparentemente confusa de caracteres, que não faz nenhum sentido e não leva a lugar nenhum. [33] As técnicas de codificação constituem uma parte importante da segurança dos dados, pois protegem informações confidenciais de ameaças que incluem exploração por malware e acesso não autorizado por terceiros. Como já mencionado, a criptografia de dados é uma solução de segurança versátil e multifuncional: pode ser aplicada a um dado específico sigiloso, como uma senha ou alguma palavra-chave, ou, mais amplamente, a todos os dados de um arquivo, como foto, vídeo, arquivo BIM (como ocorre nesse projeto) ou ainda a todos os dados contidos em alguma mídia de armazenamento, a fim de proteger arquivos em nuvem ou dados em trânsito durante a navegação na internet.[33]

Atualmente, todo órgão, empresa ou projeto, por mais simples que seja, devem estar protegidos contra ataques maliciosos, garantindo o sigilo de dados confidenciais, tanto da empresa, como dos clientes, funcionários, projetistas, etc. Nesse sentido, a criptografia de dados torna-se uma aliada importante à segurança das informações.

2.6.1 Hash

Normalmente, o uso da criptografia na proteção de senhas se faz através de uma função de hash. Nela, os dados são embaralhados de forma a mesclar caracteres para dificultar um

possível ataque. Trata-se de um algoritmo de mão única que modifica a senha para um valor distinto, de modo que se um atacante descobrir de alguma forma esse valor ele não consegue, a princípio, saber o conteúdo original da senha com facilidade.

Obter a senha a partir do hash é difícil, mas não impossível. Os métodos de ataque a segurança de dados vão se intensificando e se aperfeitçoando com a evolução computacional, e inúmeros sistemas são hackeados ao redor do mundo. Existem até sites que fazem esse serviço de quebra de hash e possuem diversos valores de hashes já quebrados. Evidentemente, a maioria dos usuários comuns da internet optam por senhas consideradas fracas, para uma melhor memorização, bem como utilizam a mesma senha em vários serviços distintos. Se toda vez que uma senha com o mesmo conteúdo fosse inserida numa função hash ela produzisse o mesmo valor, logo, a descoberta de um valor hash em qualquer uma dessas senhas poderia ser usada para atacar qualquer usuário que usasse essa senha em qualquer sistema. Baseado nessa hipótese (que é real), surge a técnica do salt.

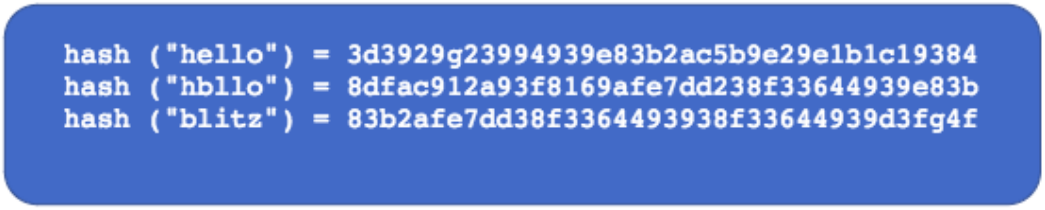

Figura 2.10 – Exemplo de palavras no processo de Hash

Fonte: Site Okta

2.6.2 SHA-256

Publicada em 2001 pelo NIST como um padrão federal dos Estados Unidos. A família SHA-2 é composta por seis funções hash com resumos de 224, 256, 384 ou 512 bits, esses números representam o tamanho das saídas para cada um desses algoritmos, em bits. A criptografia utilizada neste trabalho utilizou a função de hash SHA-256.

O preenchimento da mensagem no algoritmo de SHA-2 ocorre acrescentando um bit 1, no final da mensagem, acrescenta-se 0 até que a palavra tenha o tamanho proporcional à quantidade de blocos, e por fim concatenar-se com bits para sinalizar o tamanho da mensagem ao último bloco. A computação do resumo inicia-se com a mensagem sendo dividida em blocos de 512 bits e a inicialização das variáveis de estado. Em seguida é realizada uma série de operações lógicas descritas por funções. Essas operações são deslocamentos, XOR(ou exclusivo), AND e NOT. A mensagem de 512 bits é dividida em 16 palavras, cada uma com 32 bits é expandida para 64 palavras de 32 bits. Posteriormente, para cada palavra é realizada uma

rodada do algoritmo, mas uma particularidade do SHA-2 é que cada rodada é composta por uma constante diferente. Por fim, as variáveis de estados geradas inicialmente são somadas módulo 2 (XOR) com os valores resultante das rodadas. Em todos os algoritmos da família SHA-2, esses procedimentos são os mesmos, variando tão somente os tamanhos de 32 bits para 64 bits para o blocos de mensagem. Na família de algoritmo SHA-2, as operações são simples e possuem bom desempenho. Entretanto, o algoritmo é fortemente sequencial, não permitindo fácil paralelização. Cada rodada do algoritmo somente pode ser computada antes de cada palavra com seus então 32 bits tendo sido calculados anteriormente. Isso requer uma sequência para realizar a computação. O último bloco que contém o preenchimento para garantir que todos sejam múltiplos de 512 bits usa, como dito, as funções de preenchimento que hoje são implementadas em software que requerem custo mínimo de processamento.[38]

2.6.3 Salt

Em criptografia, salt é uma cadeia de caracteres aleatória agregada a uma operação hash. É um dado aleatório que é usado como uma entrada adicional para uma função unidirecional que "quebra"os dados, uma senha ou frase-passe. Os salts são usados para proteger as senhas, palavra-chave ou arquivo no armazenamento. Esse valor aleatório de salt gerado é guardado no banco de dados com o hash, pois ele será necessário no processo de verificação da senha, quando um usuário tentar logar com os dados já cadastrados.

O salt pode ser anexado antes ou depois dos dados de hash. Devido a essa aleatoriedade e extensa variedade de caracteres utilizados nessa técnica, seu uso impede que duas senhas iguais tenham o mesmo valor de hash. A utilização do salt permite que o hash gerado seja completamente diferente para cada vez que for gerado, mesmo que a senha seja exatamente igual. Dessa forma, mesmo que dois ou vários usuários escolham senhas idênticas para login em um sistema, cada hash será completamente diferente, pela presença do salt adicionado.

Figura 2.11 – Exemplo de palavras no processo de Hash + Salt

Fonte: Site Okta

A técnica do Salt aplica o conceito de um nonce criptográfico: um número aleatório que só é utilizado uma vez. O próprio nome, derivado do inglês, já indica o significado: N = *Number* (Número) e *Once* = Uma vez. Apesar do N do nome indicar um número, essa

definição se aplica a letras também. Portanto, trata-se exatamente do que o salt faz, um conjunto de caracteres aleatórios e únicos, a fim de evitar a duplicidade de mensagens e confundir e dificultar a ação de atacantes.

Com a aplicação de salt, as técnicas já conhecidas empregadas para descobrir senhas hasheadas, como *rainbow tables*, tabelas de *lookup* e de *lookup* reverso, se tornam impossíveis. Pela impossibilidade do atacante prever de qualquer forma os valores de salts que serão gerados ao criar os hashes de senha, a elaboração de tabelas de senhas com seus respectivos hashes, como *Rainbow Tables*, que é uma extensa lista de hashes pré-computadas para as senhas mais comumente usadas, se torna também inviável. Mais tecnicamente, salts são tão efetivos no combate a *rainbow tables* porque eles, na prática, aumentam exponencialmente a complexidade da senha.

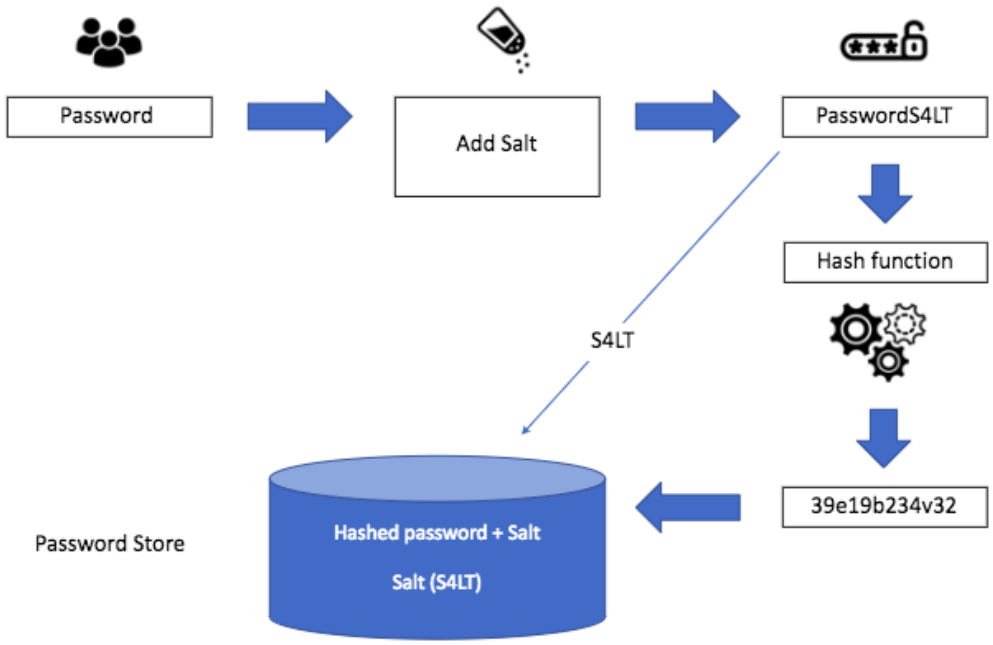

Figura 2.12 – Processo de adição do Salt

Fonte: Site Okta

Os salts podem ser criados usando bibliotecas e funções do tipo CSPRNG (*Cryptographically Secure Pseudo-Random Number Generator*), ou Gerador de Números Pseudo-Aleatórios Criptograficamente Seguros, que podem ser encontradas na maioria das linguagens de programação, como C, Java, Python, PHP, Ruby, etc.

3 IMPLEMENTAÇÃO DA PROPOSTA

3.1 METODOLOGIA

3.1.1 Descrição do software

O software desenvolvido neste trabalho foi denominado CBCAD. O CBCAD tem como propósito facilitar o trabalho do Corpo de Bombeiros no combate ao incêndio, proporcionando uma visão mais rápida e clara da estrutura em perigo. A construção do software foi separada em três partes em paralelo, são elas:
•Arquivos executáveis.
•Biblioteca em Python do CBCAD.
•*Workbench* do FreeCAD.

Os arquivos executáveis garantem que o pacote CBCAD vai estar acessível ao python e inicializa o programa. A biblioteca em Python é um wrapper utilizando o *Tkinter* (biblioteca python). Foram criadas várias classes que englobam um widget específico do *Tkinter*, utilizando de métodos para controlar características específicas como tamanho, localização, cor entre outros. Além disso, foram criadas classes específicas para cada janela do sistema como: tela de Login, tela de novo usuário e tela de abertura de arquivos. Essas classes recebem os widgets dentro dos seus métodos para assim controlar suas propriedades.

O *workbench* é composto por dois arquivos : init.py e o initgui.py. Estes são ativados sempre que o workbench for aberto na interface gráfica. Nesses arquivos é definido o comando “SFN”, responsável pela Macro do FreeCad. Esses arquivos são instalados no diretório /mod do FreeCAD.

3.1.2 Instalação do software

Para a instalação do aplicativo é necessário que o computador do usuário tenha ao menos o python 3.5 instalado. Os seguintes passos devem ser seguidos para a instalação do software:
•Baixar o arquivo “CBCAD.zip” e extrair os arquivos na pasta desejada.
•Executar o programa “*Powershell*” como administrador.
•Entrar na pasta em que os arquivos foram extraídos e executar o comando “python install.py” , se preferir no próprio diretório executar “python <to>CBCAD.py”.
•Caso já tenha uma versão do CBCAD instalada no seu computador o programa exibirá a

mensagem ”There is already a CBCAD version installed on this computer.Would you like to replace it?”, basta digitar “Y” caso queira prosseguir com a instalação.
Aguardar a instalação dos pacotes necessários.
•Caso apareça a mensagem “CBCAD was successfully installed”, o programa estará pronto para a execução.

Além dos pacotes necessários para a correta execução do aplicativo, o CBCAD baixa automaticamente a versão mais atual do “*pymysql*”, biblioteca necessária para a posterior conexão ao banco de dados. A pasta “CBCAD” estará nos arquivos de programas contendo o executável responsável pela inicialização do aplicativo. Podemos observar o exemplo de instalação bem sucedida na figura 3.1.

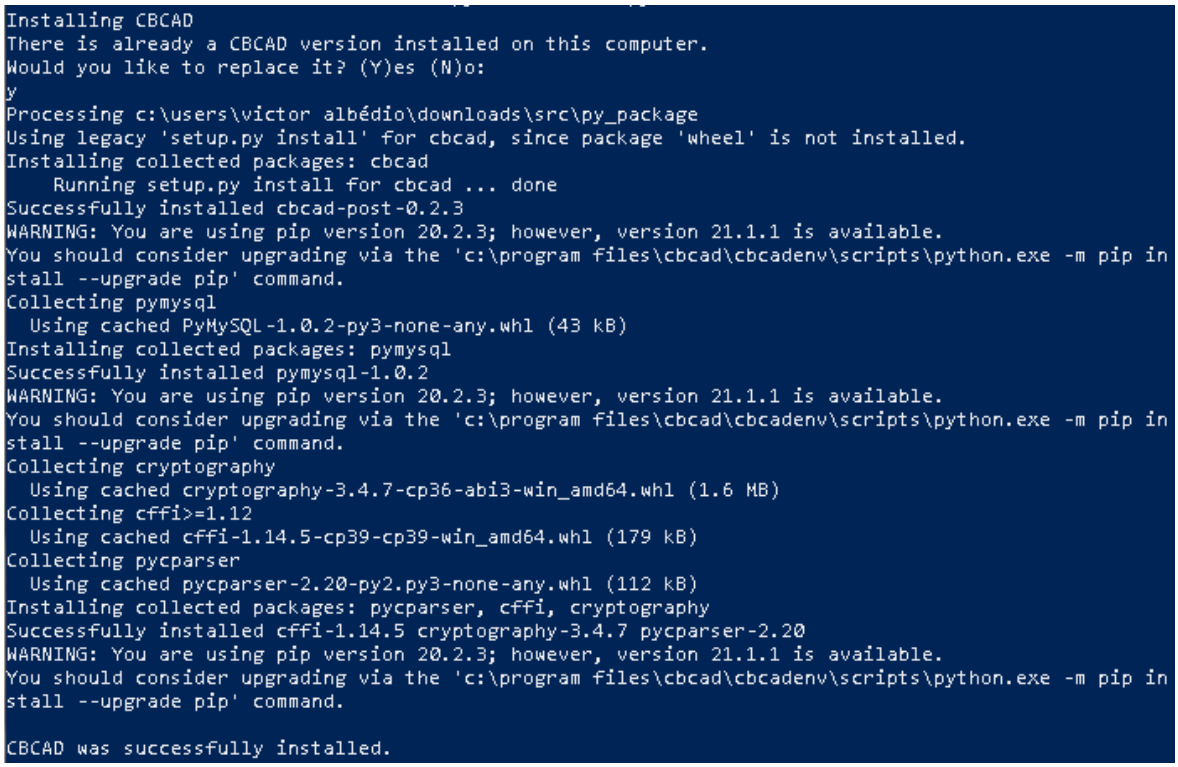

Figura 3.1 – Instalaçao do CBCAD

Fonte: Autor

3.1.3 Programas Utilizados

Utilizando Python e o software Atom para dsenvolvimento dos códigos,os programas desenvolvidos para este trabalho foram:

• sfn.py

• InitGui.py
• sensordata.py
• guielements.py
• filesearcher.py
• login.py
• cryptographwindow.py
• credentials.py
• cbcadinstance.py
• CBCAD.py
• CBCAD.pyw
• cbcadinstance.pyw
• filesearcher.pyw
• login.py

3.2 ARQUITETURA

A Figura ilustra a proposta de arquitetura de referência, para comunicação de centrais de alarme de incêndio em nuvem:

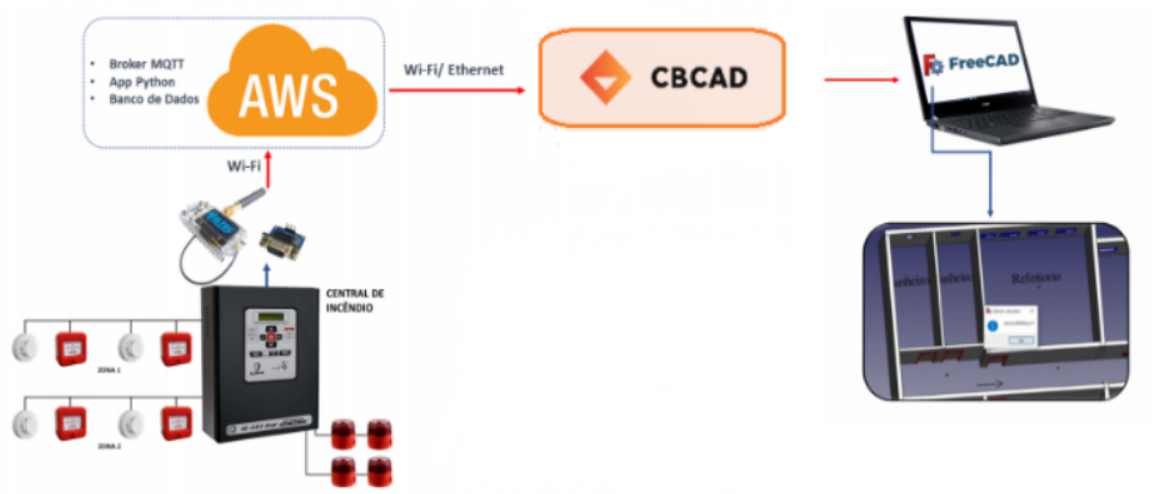

Figura 3.2 – Arquitetura do sistema

Fonte: Autores

A Central de incêndio se comunica com a Internet, e consequentemente, com a nuvem AWS, através do dispositivo ESP32 atuando com conexão Wi-Fi junto a rede. Na nuvem, existe uma máquina específica para executar o broker responsável por receber e registrar as

informações obtidas de um possível incêndio. Dentro da nuvem AWS se estabelecem as condições de segurança e persistência das informações. Assim, pode-se ter a qualquer momento o acesso às informações armazenadas em banco de dados. Após esse momento, o CBCAD entra no sistema, conseguindo extrair as informações do banco de dados e reproduzindo no FreeCAD. A imagem do notebook com o símbolo do FreeCAD, constante na figura em questão, representa o software que abre o IFC produzido pelo projeto BIM e o mostra com possibilidade de atualização das informações dos sensores em tempo real. A arquitetura proposta é simples e segura. Afinal, de acordo com as premissas deste estudo, o resultado deve ser rápido e oportuno para as ações dos bombeiros. Ainda neste contexto, vale destacar que este sistema proposto é calcado no funcionamento de um sistema de intercomunicação maior, que passa pela disponibilização de enlaces estáveis de troca de informações.[1]

3.3 FLUXOGRAMA DA SOLUÇÃO

De acordo com a validade das informações inseridas ou as decisões tomadas pelo cliente, temos o seguinte fluxograma para a proposta do software cbcad:

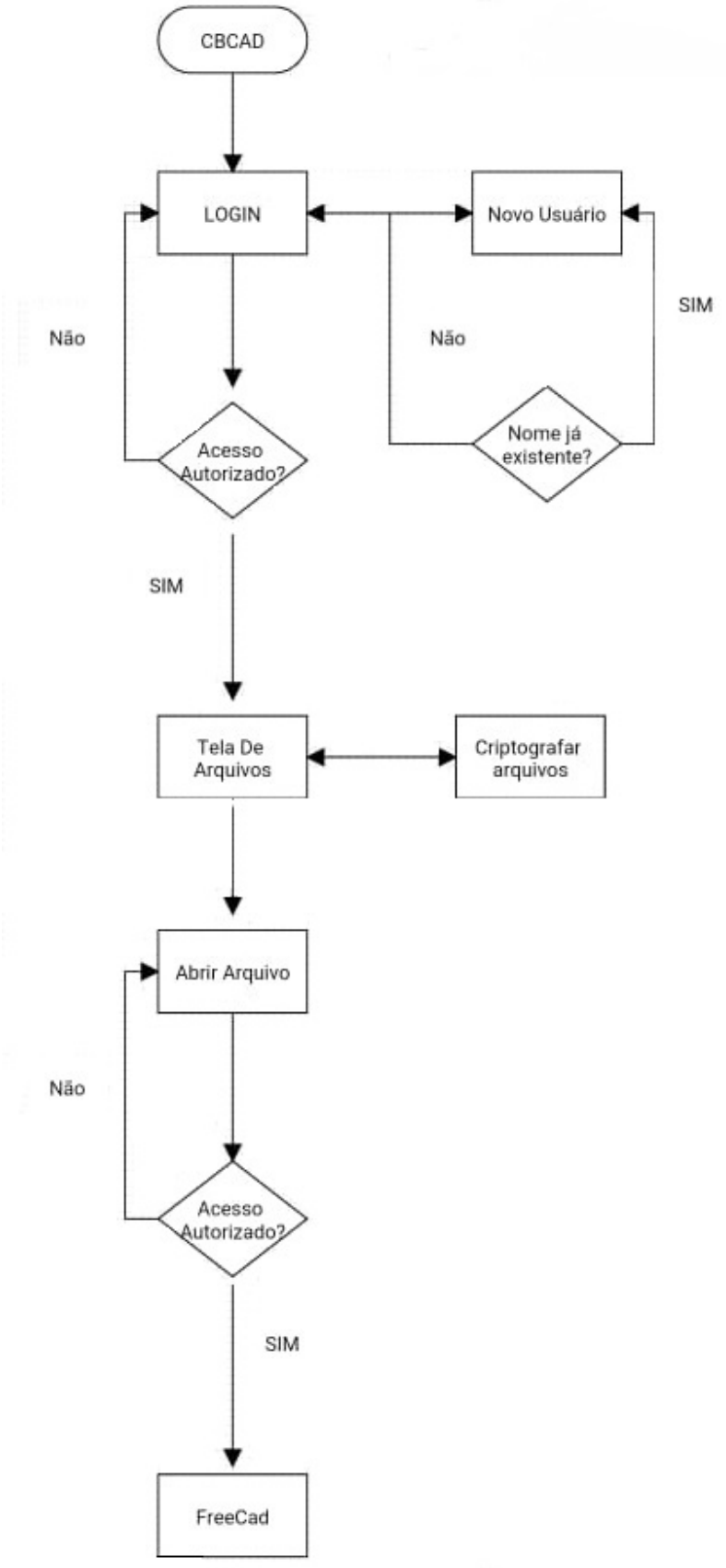

Figura 3.3 – Fluxograma da solução

Fonte: Autores.

4 RESULTADOS

4.1 TELA DE LOGIN E NOVO USUÁRIO

Ao executar o software, o programa abrirá uma tela de Login, e o novo usuário deverá cadastrar-se clicando no botão “New User”, definindo um “Username” e um “Password” para a sua conta. Caso já tenha um cadastro com o mesmo nome, o programa abre um “pop-up” solicitando ao utilizador do serviço um novo nome de usuário.Além disso, o programa também alerta caso o campo do usuário ou da senha não estejam preenchidos. As figuras 4.1, 4.2 e 4.3 mostram as telas de login e os alertas gerados pelo programa em eventuais erros ou omissão de dados.

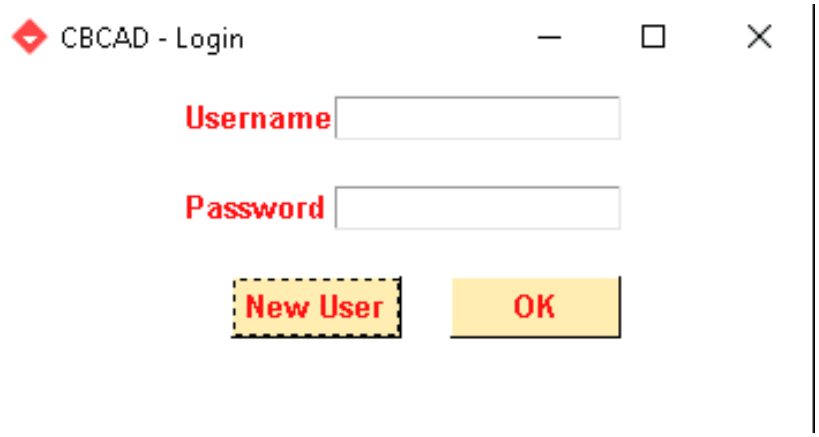

Figura 4.1 – Tela de login

Fonte: Autor

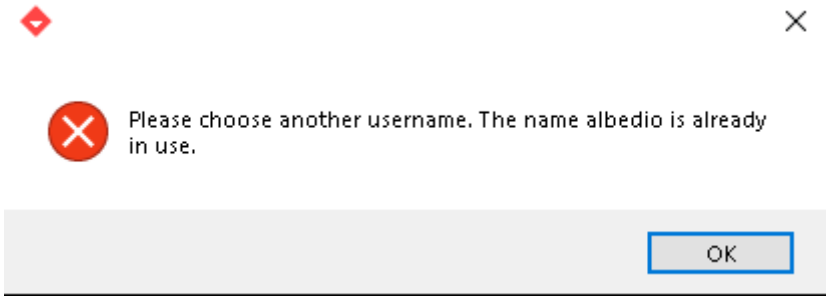

Figura 4.2 – Alerta de usuário já existente

Fonte: Autor

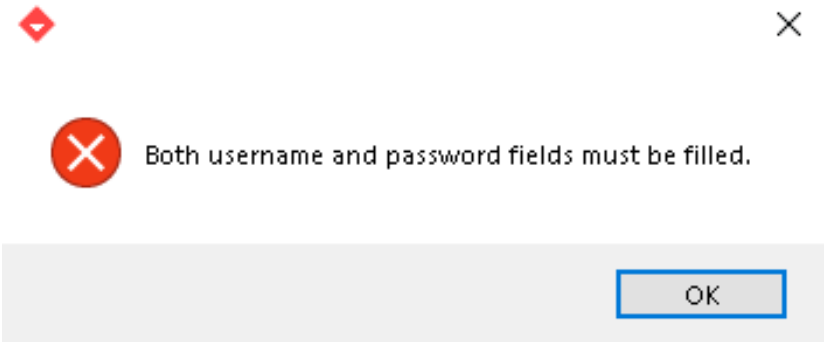

Figura 4.3 – Alerta de campos não preenchidos

Fonte: Autor

O cadastro de usuários no programa é necessário para que apenas os bombeiros autorizados possam ter acesso as plantas criptografadas. Quando definida a senha, o programa gera dois arquivos na pasta do usuário, localizada em "C:Users/NomedoUsuario/AppData/Local/-CBCAD/credentials/Username", chamadas pass e salt.

O salt nada mais é que um número gerado aleatoriamente pelo programa que é adicionada a senha dada pelo usuário antes desta ser criptografada. Desse modo, é dada mais uma aleatoriedade às senhas além de impedir que as mesmas senhas gerem um mesmo hash, dificultando assim o trabalho de atacantes.

Adicionada a senha criada é gerada uma token, esta é criptografada com a função de hash SHA-256 gerando o conteúdo da pasta “pass”. Para a verificação da senha, o programa tenta descriptografar o arquivo com a chave (senha do usuário) e o salt armazenado, caso a tentativa tenha sucesso, a senha fornecida é comparada a senha armazenada para uma segunda verificação, caso esteja tudo certo, o acesso é autorizado. O listing 4.1 é uma parte do código mostrando como foi feito o procedimento.

```
class Credentials:
    def __init__(self, username, password):
        self.username = username
        self.password = password

    def __eq__(self, other):
        if self.username != other.username:
            return False
        if self.password != other.password:
            return False

        return True

    def hash(self, salt):
        pass
```

```
def save(self):
    for credentials_dir in CredentialsList.credentials_dirs:
        if credentials_dir.username == self.username:
            raise ExistingUserError

    credentials_dir = CredentialsDir(self.username)
    credentials_dir.create()

    salt = os.urandom(16)
    credentials_dir.salt = salt

    token = FernetToken(salt, self.password)
    credentials_dir.encrypted_pass = token.encrypt(
        self.password.encode('utf-8'))

@property
def authorized(self):
    salt = self.salt
    if salt is None:
        return False

    token = FernetToken(salt, self.password)
    try:
        trial_password = token.decrypt(
            self.credentials_dir.encrypted_pass).decode('utf-8')
        if trial_password == self.password:
            return True
    except InvalidToken:
        pass

    return False

@property
def salt(self):
    try:
        return self.credentials_dir.salt
    except AttributeError:
        pass

@property
def credentials_dir(self):
    for credentials_dir in CredentialsList.credentials_dirs:
        if credentials_dir.username == self.username:
            return credentials_dir
```

Listing 4.1 – Código de criptografia das senhas

A parte do código apresentado no listing 4.1 mostra os métodos utilizados para definir

a classe "credentials". O programa gera um salt como um número aleatório de 16 bytes, criptografa a credencial “senha+salt”, e posteriormente tenta descriptografar o arquivo com a chave (senha do usuário), e por meio desse processo o software autoriza ou não o usuário de acordo com a senha fornecida.

4.1.1 Tela de abertura de arquivos

Autenticado o usuário, o programa agora abre uma nova janela apresentando as seguintes opções:

•O usuário pode selecionar a extensão do arquivo a ser pesquisado (.ifc,.cbcad).
•O usuário pode escolher o diretório padrão para a pesquisa de documentos.
•O usuário pode abrir o explorador de arquivos para a abertura de um arquivo específico.
•O usuário pode criptografar o diretório ou o arquivo desejado.

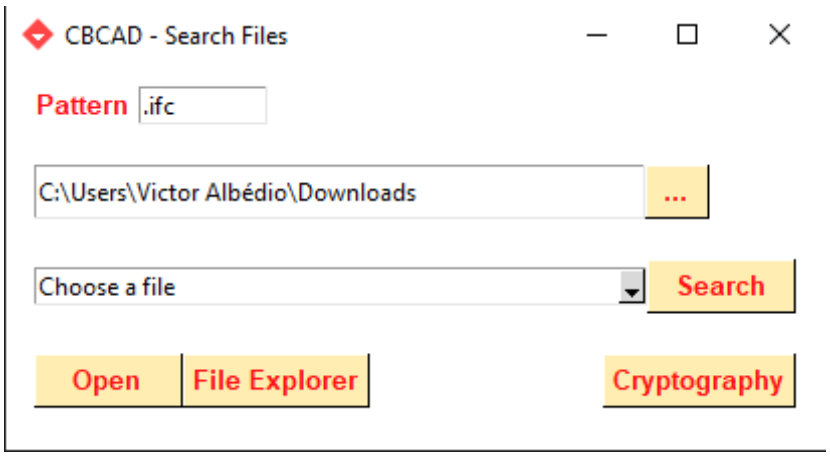

Figura 4.4 – Tela de abertura de arquivos

Fonte: Autor

O bombeiro responsável pela abertura das plantas precisa de um processo rápido, que atenda a toda requisição com a maior praticidade possível. Pensando nisso, o CBCAD disponibiliza diversas opções para a rápida abertura de arquivos. O usuário pode pesquisar por nome ou pela extensão desejada, em qualquer diretório do seu computador digitando a pasta desejada e apertando em “search”, nesse caso todos os arquivos com o nome ou extensão escritas no campo “pattern” ficarão disponíveis.Outra maneira é também pesquisar manualmente o arquivo por meio da opção “File explorer” , que abrirá o explorador de arquivo.

Outra funcionalidade do software é criptografar os arquivos desejados, os quais só poderão ser acessados por funcionários autorizados. Ao clicar no botão “ Cryptografy”, o

responsável poderá criptografar o arquivo ou o diretório desejado, para isso basta especificar o pasta de destino e apertar no botão “Encrypt”, gerando assim o "arquivo.CBCAD". A figura 4.6 demonstra o funcionamento desse sistema.

Figura 4.5 – Tela de Criptografia de arquivos

Fonte: Autor

Figura 4.6 – Exemplo de arquivo criptografado (extensão .CBCAD)

Fonte: Autor

O arquivo utiliza os mesmos procedimentos explicados na criptografia das senhas, utilizando do salt gerado por esse usuário para “salgar” o arquivo, ou seja, um arquivo ou pasta criptografados terão acesso negados caso tentem acessá-los de outro usuário.

Ao abrir o “arquivo.CBCAD” o programa automaticamente descriptografa o arquivo e inicializa o FreeCad. Caso o bombeiro logado não esteja autorizado a abrir o documento, o programa abre um pop-up de alerta sobre o acesso negado. Caso o programa tente ser aberto

sem um arquivo ou por erro no diretório, uma mensagem de erro aparecerá informando que o arquivo não existe. As telas de alerta podem ser visualizadas nas figuras 4.7 e 4.8.

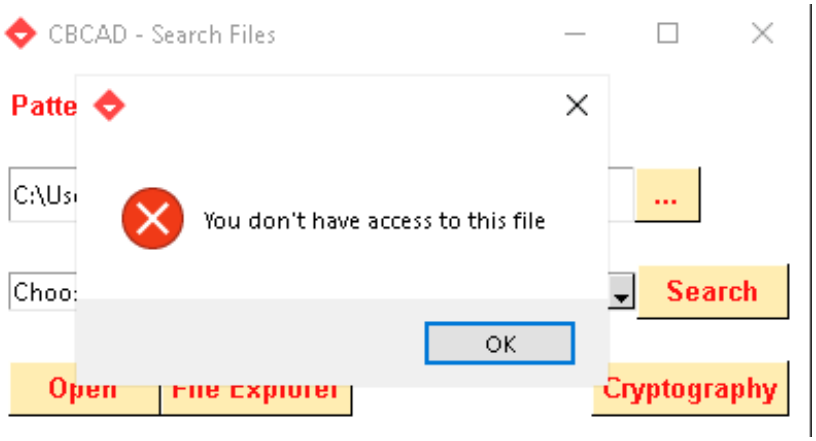

Figura 4.7 – Alerta de acesso não autorizado

Fonte: Autor

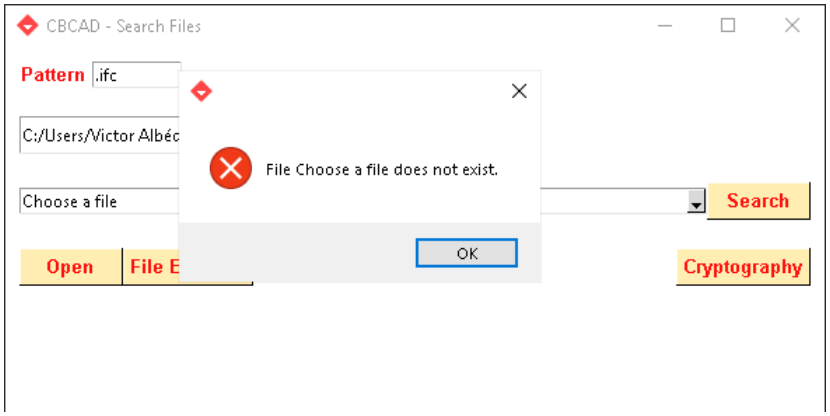

Figura 4.8 – Alerta de diretório ou arquivo não existentes

Fonte: Autor

4.2 FREECAD

A última etapa do projeto consiste em abrir o arquivo já descriptografado, modificar o FreeCad para uma interface mais limpa, conectar ao banco de dados e atualizar e informar o bombeiro em tempo real sobre o estado dos sensores do projeto.

4.2.1 Abertura e visualização do FreeCad

Mais uma melhoria do projeto foi a "limpeza"da interface de visualização do software FreeCAD. Observando e comparando as imagens, percebe-se que o FreeCAD naturalmente

oferece diversas possibilidades para editar, mudar o cursor, navegação e visualização, entre outras inúmeras funcionalidades. Para proporcionar essas opções de manuseio da plataforma, a aplicação expõe muitos ícones nas barras de cima e laterais. Como a aplicação CBCAD já automatiza processos e melhora a visualização, a maioria dos ícones acabam perdendo a sua funcionalidade e acabam por "poluir"a imagem. Pensando numa visualização mais ampla e limpa das edificações em momentos decisivos no combate a incêndios, o CBCAD automatiza o processo de excluir esses ícones das barras laterais e de cima.

O programa desenvolvido inicializa automaticamente o Freecad e abre o arquivo de uma maneira mais limpa e ágil para o responsável. Serão apresentados apenas o projeto BIM e a janela de relatórios, que será utilizada para mostrar os dados dos sensores. Além disso foi criado um módulo, o qual deverá ser executado para que a Macro faça conexão ao banco de dados. O listing 4.2 mostra o programa responsável pelas modificações no Freecad.

```
from PySide import QtGui
from pathlib import Path
import os
#
    ==========================================================================

program_files_dir = Path(os.environ['ProgramFiles'])
CBCAD_dir = program_files_dir / 'CBCAD'
with open(CBCAD_dir / 'freecad_dir.txt', 'r') as file:
    FreeCAD_dir = Path(file.read())
mod_dir = FreeCAD_dir / 'Mod'
workbench_dir = mod_dir / 'CBCAD'
resources_dir = workbench_dir / 'resources'
#
    ==========================================================================

class CBCAD(Workbench):
    global QtGui
    global resources_dir
    MenuText = "CBCAD"
    ToolTip = "CBCAD"
    Icon = str(resources_dir / 'icons/cbcad.svg')

    def Initialize(self):
        from sfn import SFN

        Gui.addCommand('SFN', SFN())
        self.list = ['SFN']
        self.toolbar_name = "CBCAD Commands"
        self.appendToolbar(self.toolbar_name, self.list)

    def Activated(self):
```

```
        self.hide_panels()
        self.hide_toolbars()
        FreeCADGui.runCommand('SFN', 0)

    def hide_panels(self):

        mainWindow = FreeCADGui.getMainWindow()
        dockWidgets = mainWindow.findChildren(QtGui.QDockWidget)
        for dw in dockWidgets:  # Report view is excluded now
            if dw.objectName() in ["Python console", "Combo View"]:
                dw.hide()

    def hide_toolbars(self):
        main_window = FreeCADGui.getMainWindow()
        toolbars = main_window.findChildren(QtGui.QToolBar)
        for toolbar in toolbars:
            name = toolbar.objectName()
            if name != self.toolbar_name and name != 'Workbench':
                toolbar.hide()

    def Deactivated(self):
        return

    def ContextMenu(self, recipient):
        self.appendContextMenu("CBCAD", self.list)

    def GetClassName(self):
        return "Gui::PythonWorkbench"


Gui.addWorkbench(CBCAD())
```

Listing 4.2 – Código de criação do workbench

O programa realiza a *workbench* na pasta "/Mod" dentro do FreeCad. O próximo passo foi definir a classe *workbench*, alguns métodos utilizados foram customizados e outros não. Para melhor visualização de seu funcionamento e entendimento do programa, os métodos serão divididos em etapas:

• *GetClassName*: Essa classe é necessária para o uso do Python na definição do *workbench*;

• *ContextMenu*: Responsável pelo botão no módulo dentro do FreeCad;

• *Initialize*: Define como será apresentado o *workbench* ao usuário no e define os comando SFN, esse método é ativado logo ao abrir o software;

• *Activated*: Acionado ao usuário entrar no módulo "CBCAD" chama os métodos "*Hide_panel*" e "*Hide_toolbar*" escondendo os painéis de visualização para uma melhor visualização do programa. Além disso, o comando definido no "*Initialize*" é executado, chamando a classe

"SFN"que fará a conexão com o banco de dados.

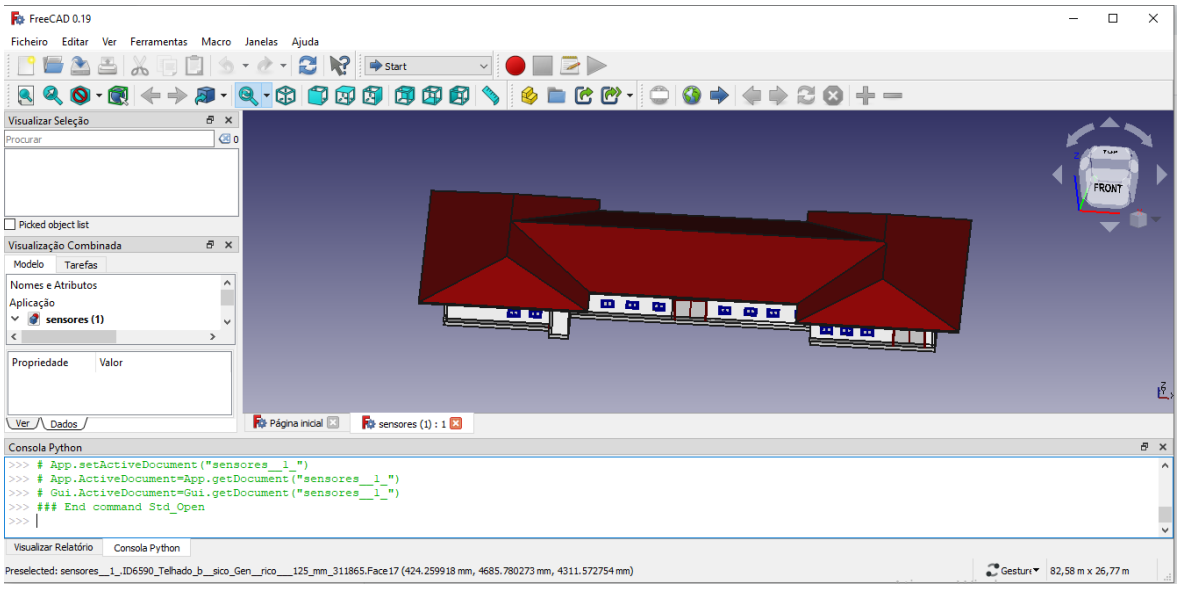

Figura 4.9 – Visualização do arquivo no Freecad normalmente

Fonte: Autor

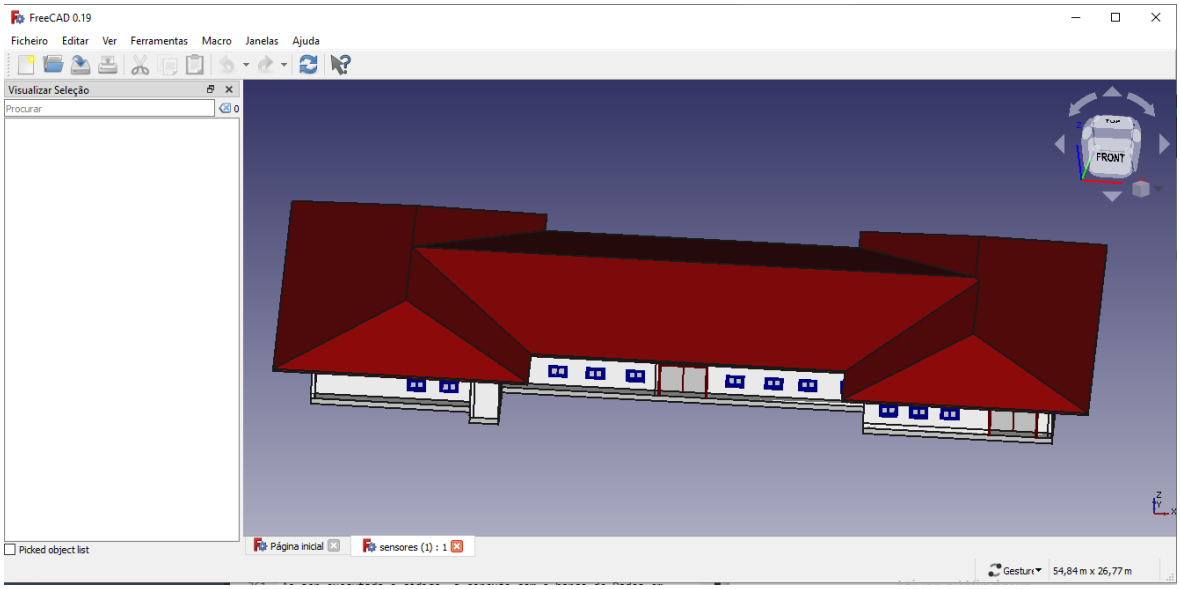

Figura 4.10 – Visualização do arquivo no Freecad com o CBCAD.

Fonte: Autor

4.3 CONEXÃO AO BANCO DE DADOS

Ao ser executada o código, a conexão com o banco de Dados em nuvem é feita, a cada 20 segundos o programa verifica o estado de cada sensor do projeto. O sensor, ao ser ativado, muda de cor e de tamanho a fim de uma observação mais nítida dos componentes em alerta. Além disso, em sua primeira ativação um *pop-up* é ativado mostrando data, hora e identificação do detector ativo. Ao fechar as janelas, as informações ficam disponíveis na aba "visualizar relatórios" e podem ser acessadas novamente caso um outro sensor seja selecionado pelo usuário. O listing 4.3 mostra o programa responsável pela conexão ao banco de dados e alteração dos sensores.

```
import FreeCAD
import FreeCADGui
from PySide import QtCore, QtGui
import Draft
import sys
from pathlib import Path
import os

program_files_dir = Path(os.environ['ProgramFiles'])
CBCAD_dir = program_files_dir / 'CBCAD'
cbcad_env_packages = CBCAD_dir / 'cbcadenv/Lib/site-packages'
sys.path.append(cbcad_env_packages)

from pymysql import connect, err


with open(CBCAD_dir / 'freecad_dir.txt', 'r') as file:
    FreeCAD_dir = Path(file.read())
mod_dir = FreeCAD_dir / 'Mod'
workbench_dir = mod_dir / 'CBCAD'
resources_dir = workbench_dir / 'resources'


#
    ================================================================================

class SFN:
    global resources_dir

    def __init__(self):
        self.active_sensors = {}
        self.selected_obj = None

    def GetResources(self):
        return {'Pixmap': str(resources_dir / 'icons/cbcad.svg'),
```

```
            'MenuText': "Secure Fire Net",
            'ToolTip': "Check the CB database for active sensors"}

    def Activated(self):
        FreeCAD.Console.PrintMessage(
            'Starting connection with the CB database\n')

        timer = QtCore.QTimer()
        timer.timeout.connect(self.update)
        timer.start(20000)
        self.update_timer = timer

        timer = QtCore.QTimer()
        timer.timeout.connect(self.inform)
        timer.start(2000)
        self.inform_timer = timer

        FreeCAD.Console.PrintMessage(
            'Verifying database every 20 seconds....\n')

    def inform(self):

        sel = FreeCADGui.Selection.getSelection()
        obj = sel[0]

        if obj is not self.selected_obj:
            for sensor, info in self.active_sensors.items():
                if sensor is obj:
                    self.print_info(*info)

        self.selected_obj = obj

    def print_info(self, obj_label, data, endereco):

        FreeCAD.Console.PrintMessage('Sensor: ' + obj_label + '\n')
        FreeCAD.Console.PrintMessage("Data: " + data + '\n')
        FreeCAD.Console.PrintMessage("Endereco: " + endereco + '\n')
        FreeCAD.Console.PrintMessage("\n\n")

    def update(self):

        try:
            db_connection = connect(host='3.23.51.225',
                                    user='db_remote_admin',
                                    password='123',
                                    database='central_incendio')
```

```
except err.OperationalError:
    FreeCAD.Console.PrintMessage('Connection␣Error.'
                                 '␣Trying␣again␣soon.')
    return

cursor = db_connection.cursor()
sql = "SELECT␣id,␣nome,␣valor,␣timestamp␣FROM␣centrais"
cursor.execute(sql)

for ID, nome, valor, timestamp in cursor:
    endereco = valor[1:5]
    laco = valor[5:6]
    evento = valor[6:8]
    hora = valor[8:10]
    minuto = valor[10:12]
    dia = valor[12:14]
    mes = valor[14:16]
    ano = valor[16:18]

    obj_label = "sensor" + endereco + "laco" + laco
    data = (hora + ":" + minuto + "␣" + dia + "/"
            + mes + "/" + ano)

    for obj in FreeCAD.ActiveDocument.Objects:

        if (obj_label == obj.Label
                and obj not in self.active_sensors):
            self.active_sensors[obj] = [obj_label, data, endereco
                ]
            info = [obj_label, data, endereco]

            Draft.scale(obj, scale=FreeCAD.Vector(20, 20, 20),
                        center=obj.Shape.BoundBox.Center)

            obj.addProperty('App::PropertyString', 'Data')

            FreeCADGui.getDocument(
                FreeCAD.ActiveDocument.Name).getObject(
                obj.Name).ShapeColor = (0.00, 1.0, 0.00)

            obj.Data = data
            message = 'Sensor:␣' + obj_label + '\nEndereco:␣' \
                      + endereco + '\nData:␣' + data
            QtGui.QMessageBox.information(None, 'Sensor␣Ativado',
                                          message)
```

Listing 4.3 – Código de conexão ao banco de dados e alteração dos sensores

O código apresentado no listing 4.3 foi feito a partir de uma Macro anteriormente desenvolvida pelo Mestre Calvin Mariano em sua dissertação de mestrado.[1]

No código ‘sfn.py’(listing 4.3) é definida a classe “SFN”, para melhor entendimento do programa seus métodos mais importantes foram separados em tópicos, são eles:

• *GetResources*: Define a aparência e apresentação do módulo para o usuário dentro do FreeCad.
• *Activated*: Executado ao apertar o botão, esse método cria dois timers , estes podem ser definidos como "instâncias"do Python sendo executadas paralelamente ao programa. O primeiro Timer chama a função “*update*” a cada 20 segundos, a qual é responsável pela conexão ao banco de dados e atualização dos sensores, mudando sua cor e tamanho, além de mostrar a data, hora e nome do sensor ativado. O segundo timer é ativado a cada 2 segundos e chama a função “inform”, essa função verifica o objeto selecionado pelo usuário e caso este esteja ativado, suas informações são apresentadas na tela.

É importante ressaltar que a leitura do banco de dados a cada 20 segundos é essencial, pois o bombeiro pode observar o alastramento do fogo no edifício, o que é crucial para estratégias de combate ao fogo e o acompanhamento do nível de perigo apresentado pelo desastre. A figura 4.11 mostra o exemplo de uma visualização do projeto com dois sensores ativos.

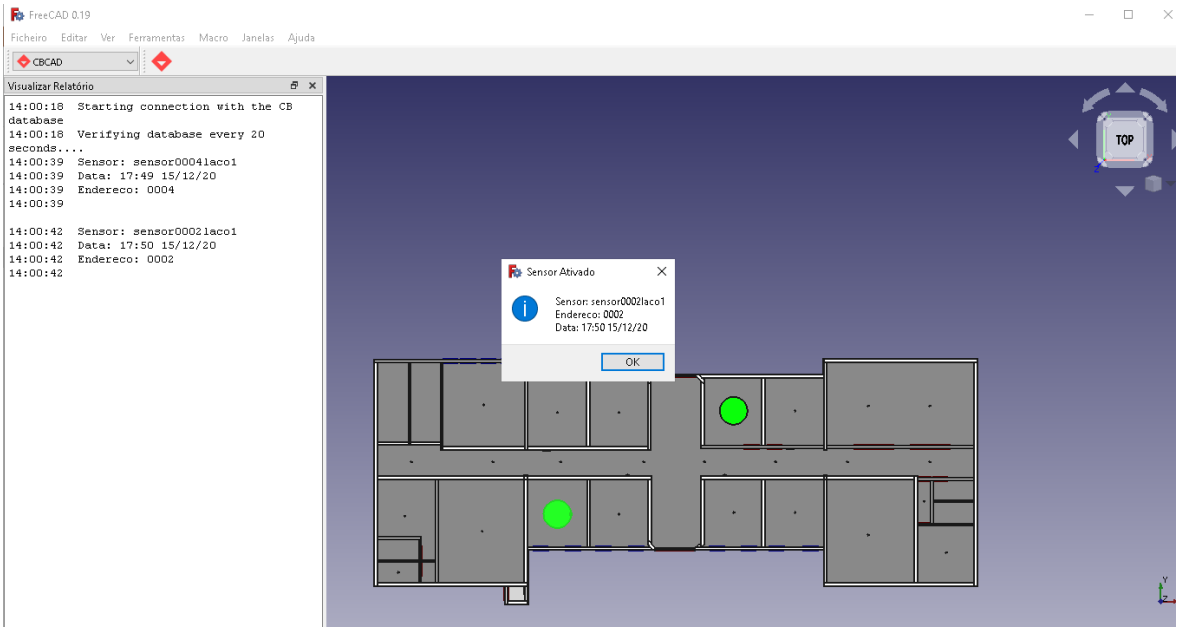

Figura 4.11 – Sensores ativos

Fonte: Autor

O CBCAD apresentou resultados positivos, da abertura do projeto até a apresentação dos sensores se passam cerca de 2 minutos e 15 segundos em uma máquina com o processador Intel Core i5 e 8GB de memória RAM. Utilizando como exemplo o 2º Grupamento de Bombeiros Militar, localizado em Taguatinga Sul, que apresenta máquinas com 8Gb e processador Intel i7, espera-se resultados efetivos com o uso do CBCAD. Realizar testes do "CBCAD" nos computadores da administração são eventos necessários, mas devido aos problemas trazidos pela COVID 19, só poderão ser realizados após o fim da pandemia. O sistema desenvolvido adiciona criptografia aos arquivos e pastas dos modelos "IFC" , buscando desenvolver não só facilidade no manuseio das informações, mas também medidas de segurança.

5 CONCLUSÃO

O trabalho buscou desenvolver de uma forma mais prática e específica a visualização de incêndios em edifícios, buscando facilitar e aprimorar o atendimento emergencial a esses casos. À medida que a complexidade dos edifícios aumenta, a dificuldade enfrentada pelas equipes de combate aumenta, sendo necessário medidas mais eficazes para evitar um desastre ainda maior. O Freecad é um software de código aberto, o que possibilitou a modificação do programa visando a facilitar o trabalho da corporação. Por fim, conclui-se que o software seria de grande auxílio a administração pública, pois traz uma ajuda fundamental ao Corpo de Bombeiros de forma prática, fácil e ágil em qualquer tipo de edifício,o que pode vir a salvar vidas e patrimônio em um eventual desastre.

Observando as recomendações e sugestões da tese de mestrado na qual esse projeto se baseou, concluímos que os objetivos propostos foram atingidos. Calvin, em sua tese, cita a necessidade de melhorar os aspectos realcionados ao FreeCAD. É notável isso no trecho: "Um dos problemas encontrados no desenvolvimento do trabalho é a capacidade limitada de leitura de arquivos IFC que o FreeCAD tem. É possível desenvolver melhoras para o aplicativo, pois ele é de código aberto.". Calvin reforça ainda mais essa necessidade em sua conclusão: que "Quanto ao software de leitura e processamento do IFC (FreeCAD), observou-se a necessidade do desenvolvimento de uma solução enxuta e específica para o caso em estudo. Ou seja, o FreeCAD tem vários módulos frutos de estudos e implementações da comunidade que devem ser retirados de forma a deixar o sistema ágil e com baixo custo computacional."Acrescentado a esse comentário, o autor cita como possíveis projetos futuros para colaborar com o seu sistema as seguintes contribuições:

• A implantação e teste do sistema em centros de operações.
• Complemento do FreeCAD para torna-lo mais eficiente na execução de arquivos IFC.
• Implantação de uma aplicação em nuvem para visualização dos projetos BIM.
• Análise do aproveitamento das informações para perícia.
• Testes de vulnerabilidades da arquitetura desenvolvida.

A projeto do CBCAD oferece algumas das demandas aludidas. Em alguns pontos, o CBCAD entrega totalmente o que foi sugerido. Em outros, o CBCAD contribui de maneira efetiva, possibilitando e até incentivando possíveis contribuições para esse sistema tão inovador e promissor.

Por fim, algumas melhorias a serem implementadas futuramente são:

1. Instalação da versão do FreeCad utilizada pelo sistema já na instalação do CBCAD.

2. Teste do sistema em um centro de operações.
3. Eliminação automática de módulos desnecessários de FreeCad para um sistema mais leve.
4. Ícone CBCAD na área de trabalho para uma execução mais rápida do software.
5. Teste de vulnerabilidades do sistema.

6 BIBLIOGRAFIA

1. CRISPIM, Calvin Mariano Rêgo. Proposta de Arquitetura Segura de Centrais de Incêndio em Nuvem. Tese (mestrado) - Universidade de Brasília, 2020.

2. AZHAR, S. Building information modeling (bim): Trends, benefits, risks, and challenges for the aec industry. Leadership and management in engineering, American society of civil engineers, v. 11, n. 3, p. 241–252, 2011.

3. FROESE, Thomas FISCHER, Martin GROBLER, F.. Industry foundation classes for project management - A trial implementation. Electronic Journal of Information Technology in Construction. 4. 17-36.1999.

4. THOMÉ, B. B. O que é BIM? Entenda agora o conceito e suas aplicações, 2020. Disponível em: <sienge.com.br/blog/voce-sabe-o-que-e-bim-entenda-o-conceito-e-suas-aplicacoes/>

5. AUTODESK. BIM and Cost Estimating, 2007.

6. BUILDINGSMART. The International Home of BIM. Disponível em: https://www.buildingsmart.org/

7. THEIN, Volker. Industry Foundation Classes (IFC) - BIM Interoperability Through a Vendor-Independent File Format. Bentley, 2011.

8. FALCK, D. COLLETE B. FreeCad [How To] - Solid Modeling with the power of the Python, 2012.

9. THOMPSON, D. B.. e-Construction: Don't get soaked by the next wave. Construction Law Briefing Paper, 2001.

10. THOMPSON, D. B., and Miner, R. G. (2007). "Building information modeling—BIM: Contractual risks are changing with technology, 2007.

11. ROSENBERG, T. L.Building information modeling, 2007.

12. PROCESSOBIM. Perfil ProcessoBIM na plataforma Pinterest. Disponível em: https://br.pinterest.com/processobim/bim-pinterest/

13. CHAVES, Hugo. O que é o formato IFC e como ele funciona?, 2020. Disponível em: https://neoipsum.com.br/o-que-e-ifc-e-como-ele-funciona/.

14. GMARTINI, Egenharia. BIM e as políticas públicas do Brasil.
Disponível em: https://www.gmartiniengenharia.com/single-post/2018/03/10/bim-e-as-politicas-públicas-do-brasil

15. RENZETTI, Arquitetura e Construção. Projeto BIM, resultados de excelencia, da criação do projeto a execução da obra, 2020. https://www.renzeti.com.br/blog/projeto-bim-resultados-de-excelencia-da-criacao-do-projeto-a-execucao-da-obra-30

16. MACHADO F, MALPICA N, BORROMEO S. .Parametric CAD modeling for open source scientific hardware: Comparing OpenSCAD and FreeCAD Python scripts, 2019.

17. VONKROGH, G.. "INTELLIGENCE Open-source Software Development, 2003.

18. MAJCHER, Janusz. Terms in BIM you should know – update. Bim Corner, 2021.

19. LERNER, Josh, and TIROLE, Jean. The Economics of Technology Sharing: Open Source and Beyond. Journal of Economic Perspectives, 19 (2): 99-120. 2005.

20. MCGRAW-HILL, Construction. Building information modeling: Transforming design and construction to achieve greater industry productivity, 2008.

21. KHEMLANI, L., PAPAMICHAEL, K., and HARFMANN, A. The potential of digital building modeling, 2006.

22. FIREBEE,ABNT NBR ISO 7240 – Norma de Produtos.
Disponível em: https://firebee.com.br/alarmedeincendiosemfio/abnt-nbr-iso-7240-norma-de-produtos/

23. PYTHON. Python.org,PEP 249 – Python Database API. Specification v2.0. Disponível em: https://www.python.org/dev/peps/pep-0249/

24. PYPI,PyMySQL. Disponível em: https://pypi.org/project/PyMySQL/

25. PYMYSQL, PyMySQL's documentation.
Disponível em:https://pymysql.readthedocs.io/en/latest/index.html

26. ORACLE. WhyMySql,Why MySQL?.Disponível em: https://www.mysql.com/why-mysql/

27. MAJCHER, Janusz. Everything worth knowing about the IFC format. Bim Corner, 2019.

28. FUGAS, Konrad. Explaining Information Requirements in ISO 19650. Bim Corner, 2021.

29. Cryptography,Disponível em: //cryptography.io/en/latest/

30. FREECADWEB. Por onde começar. Disponível em: https://wiki.freecadweb.org/Getting-started/pt

31. Tyran,Cryptography.Dispnível em: https://github.com/pyca/cryptography

32. Cryptography.
Disponível em: https://python-guide-pt-br.readthedocs.io/ptBR/latest/scenarios/crypto.html

33. ALEXANDER, Steven,passwords matter, 2012.
Disponível em: http://bugcharmer.blogspot.com/2012/06/passwords-matter.html

34. ISO 16739:2013, Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries. Disponível em: https://www.iso.org/standard/51622.html

35. ISO 16739-1:2018, Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries — Part 1: Data schema. Disponível em: https://www.iso.org/standard/70303.html

36. IFC,O que é o formato IFC e como ele funciona. Disponível em: https://neoipsum.com.br/o-que-e-ifc-e-como-ele-funciona/

37. TUDO SOBRE BIM, 2020.Disponível em :https://thorusengenharia.com.br/o-que-e-bim/

38. OLIVIER, Gracielle Forechi,Estudo e implementação do algoritmo de resumo criptográfico SHA-3,2013.