

Modelação de Propriedades de Estruturas em Modelos BIM

JOÃO PEDRO MORAIS DE SÁ

Dissertação submetida para satisfação parcial dos requisitos do grau de
MESTRE EM ENGENHARIA CIVIL — ESPECIALIZAÇÃO EM CONSTRUÇÕES

Orientador: Professor Doutor João Pedro Poças Martins

Co-Orientador: Professor Doutor João Filipe Meneses Espinheira Rio

JUNHO DE 2014

MESTRADO INTEGRADO EM ENGENHARIA CIVIL 2013/2014

DEPARTAMENTO DE ENGENHARIA CIVIL

Tel. +351-22-508 1901

Fax +351-22-508 1446

✉ miec@fe.up.pt

Editado por

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Rua Dr. Roberto Frias

4200-465 PORTO

Portugal

Tel. +351-22-508 1400

Fax +351-22-508 1440

✉ feup@fe.up.pt

🌐 <http://www.fe.up.pt>

Reproduções parciais deste documento serão autorizadas na condição que seja mencionado o Autor e feita referência a *Mestrado Integrado em Engenharia Civil - 2013/2014 - Departamento de Engenharia Civil, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal, 2014.*

As opiniões e informações incluídas neste documento representam unicamente o ponto de vista do respetivo Autor, não podendo o Editor aceitar qualquer responsabilidade legal ou outra em relação a erros ou omissões que possam existir.

Este documento foi produzido a partir de versão eletrónica fornecida pelo respetivo Autor.

À memória do meu Tio António Fernando Cavaleiro de Carvalho
e do meu Amigo e colega de curso Ricardo Filipe da Silva Correia

*"Happiness does not come from doing easy work but from the afterglow of satisfaction that
comes after the achievement of a difficult task that demanded our best."*

Theodore Isaac Rubin

AGRADECIMENTOS

A realização da presente dissertação foi apenas possível devido à presença de um conjunto de pessoas, cujo apoio se mostrou crucial não só pela partilha de conhecimentos e pelos incentivos para fazer mais e melhor, mas também por todo o sentido de motivação demonstrado. Apesar da elaboração de um documento deste tipo estar associada a um processo solitário, toda a ajuda disponibilizada atenuou com certeza esta afirmação.

Agradeço...

Ao meu orientador, Prof. Doutor João Pedro da Silva Poças Martins, pela dedicação, orientação, apoio ao longo de todas as fases deste projeto, sugestões propostas, conhecimentos transmitidos e críticas construtivas.

Ao meu co-orientador, Prof. Doutor João Filipe Meneses Espinheira Rio, por todas as manhãs e tardes disponibilizadas de apoio ao desenvolvimento deste documento, pelo espírito crítico, perfeccionismo, incentivos, atenção e sábios conselhos.

Ao Prof. Doutor José Miguel de Freitas Castro, pelo primeiro contacto com a programação em *Python*, pelo seu sentido entusiasta na utilização de aplicações informáticas aplicadas à Engenharia Civil e por todo o apoio à implementação do *Web service*.

Ao Mestre Rui Manuel Santos Barros por tornar possível a transposição da ferramenta produzida para um *Web service*.

Ao Mestre Gustavo Pinho Oliveira por toda ajuda crucial em torno da linguagem programática *Python*.

Ao Mestre Sérgio Miguel Ferreira de Pinho pela apresentação de conceitos fundamentais para uma primeira abordagem ao IFC e pelo modelo IFC disponibilizado e utilizado nesta dissertação.

À minha família, particularmente à minha Mãe, ao meu Pai e à minha Irmã pelo apoio incondicional durante todos estes anos de formação académica e pessoal e à minha namorada Ana Lúcia, por todo o acompanhamento ao longo do trabalho, pelo interesse revelado, sugestões partilhadas e sobretudo pelo apoio incondicional demonstrado.

A todos os meus amigos e colegas de curso que me acompanharam no meu percurso académico.

A todos, sem exceção, um muito obrigado.

RESUMO

A presente dissertação tem como objetivo avaliar a possibilidade de utilizar o formato aberto do BIM, o IFC, para constituição de ferramentas que permitam a obtenção automática de quantidades de elementos estruturais, disponibilizadas remotamente através de um navegador da internet ou dispositivo móvel.

Este tema insere-se no âmbito das Tecnologias da Informação e Comunicação (TIC) e na sua importância na colaboração em tarefas-tipo da Engenharia Civil.

Os métodos empregues para o seu desenvolvimento baseiam-se na aplicação de ferramentas BIM (*Autodesk Revit*, *Tekla Structures* e *Graphisoft ArchiCAD*) para modelação de estruturas (compostas por vigas, pilares, lajes e fundações diretas), associando diferentes propriedades geométricas e materiais, nomeadamente betões. Os modelos concebidos são exportados/ gravados como ficheiros IFC-SPF, de modo a avaliar a possibilidade da sua utilização como suporte de tarefas de extração de quantidades. Segue-se o desenvolvimento de rotinas para a sua extração automática, recorrendo para tal à ferramenta programática *Python*. Existem diferentes projetos referentes a ferramentas que permitem a manipulação das bases de dados de conhecimento do IFC, cujas API's (*Application Programming Interface*) são disponibilizadas para utilização por terceiros. Através do programa idealizado é possível o desenvolvimento de um *Web service* que permita o acesso remoto aos seus serviços por parte de qualquer utilizador e a partir de diversas plataformas. Pretende-se que os dados de passagem do programa sejam baseados em estruturas XML para posterior manipulação via Microsoft Office Excel, entre outros.

O tema das medições na construção é composto por inúmeras condicionantes que devem ser alvo de estudos detalhados através da constituição de casos de estudo em ferramentas BIM e, posterior, análise via IFC. Cada ferramenta BIM apresenta estruturas bem definidas e distintas para a conversão da informação para o IFC e, por isso, o mesmo modelo BIM pode gerar ficheiros IFC distintos. A quantidade de informação acoplada num modelo BIM tem influência direta no tamanho do ficheiro IFC, sendo, portanto, o poder de cálculo da ferramenta idealizada tanto mais eficiente quanto menor o tamanho do ficheiro. Com efeito, os modelos utilizados para extração de quantidades devem ter por base, numa primeira abordagem, modelos estruturais em detrimento de modelos arquitetónicos, ou seja, modelos com nível de detalhe inferior.

PALAVRAS-CHAVE: BIM, IFC, Medições na Construção, API, *Web service*.

ABSTRACT

This dissertation aims to evaluate the possibility to use the open format of BIM, IFC, for the establishment of tools that allow the obtainment of structural elements quantities in an automatically way, by providing its remote access through a web browser or other mobile devices.

This topic is inserted into the Information and Communications Technologies (ICT) and in its importance in the collaboration in Civil Engineering tasks.

The methods used for its development are based in the application of BIM tools (Autodesk Revit, Tekla Structures and Graphisoft ArchiCAD) for modelling of structures (formed by beams, columns, slabs and footings), associating different geometric properties and materials, namely concretes. The conceived models are exported/ saved as IFC-SPF files, to evaluate the possibility of its use as tasks support, in the extraction of quantities. Follows the development of routines for its automatic extraction, through the programmatic tool Python. There are different projects regarding tools that allow the manipulation of data bases of knowledge of IFC, whose API's (Application Programming Interface) are available for the use of a "third person". Through the idealized program it is possible the development of a Web service that allows the remote access to its services by any user and starting from several platforms. It is intended that the output of the program is based in XML structures for its subsequent manipulation through Microsoft Office Excel or others.

The topic of the measurements in construction is composed by countless conditioning factors that should be the aim of detailed studies, through the establishment of cases of study in BIM tools and, subsequently, in its analysis via IFC. Each BIM tool presents well defined and distinguished structures to convert to the IFC format and, consequently, the same BIM model can generate different IFC files. Moreover, the amount of information coupled in a BIM model has direct influence in the size of the IFC file. Therefore, the power of calculation of the tool idealized is so more efficient as the file size is much smaller. With effect, in a first approach to this problem, the models used for extraction of quantities should be based in structural models instead of architectonic models, in other words, models with less details.

KEYWORDS: BIM, IFC, Measurements in Construction, API, Web service.

ÍNDICE GERAL

AGRADECIMENTOS.....	I
RESUMO	III
ABSTRACT.....	V
1 INTRODUÇÃO.....	1
1.1. ENQUADRAMENTO	1
1.2. ÂMBITO E OBJETIVOS	2
1.3. ESTRUTURA DA DISSERTAÇÃO	3
2 ESTADO DA ARTE.....	5
2.1. INTRODUÇÃO.....	5
2.2. PROBLEMAS E SOLUÇÕES NA INDÚSTRIA AEC.....	5
2.3. BIM	7
2.3.1. CONCEITO	7
2.3.2. BREVE HISTÓRIA	10
2.3.3. VANTAGENS E DESVANTAGENS DO MODELO VIRTUAL.....	12
2.3.4. IMPLEMENTAÇÃO	15
2.4. REGRAS DE MEDIÇÃO	19
2.5. BIM NO CONTEXTO DE MEDIÇÕES.....	20
2.6. IFC.....	21
2.6.1. CONCEITO.....	21
2.6.2. BREVE HISTÓRIA	23
2.6.3. ESTRUTURA/ ARQUITETURA DO IFC.....	23
2.7. WEB SERVICE.....	25
2.7.1. CONCEITO	25
2.7.2. WEB SERVICE DO TIPO “RESTFUL”	25
2.7.3. VANTAGENS	27
3 EXTRAÇÃO DE QUANTIDADES SUPOSTADA POR BIM.....	29
3.1. DESCRIÇÃO GERAL	29
3.2. REGRAS DE MEDIÇÃO	29
3.3. IFC (<i>INDUSTRY FOUNDATION CLASSES</i>).....	31

3.3.1. LINGUAGEM E SIMBOLOGIA	31
3.4. CASOS DE ESTUDO	33
3.4.1. ENQUADRAMENTO GERAL.....	33
3.4.2. ISM - <i>INTEGRATED STRUCTURE MODELLING</i>	35
3.5. MODELOS ANALÍTICOS PARA IFC 2x3 E IFC 4.....	37
3.5.1. IDENTIFICAÇÃO DE MATERIAIS.....	38
3.5.2. IDENTIFICAÇÃO DE ELEMENTOS.....	39
3.5.3. ASSOCIAÇÃO DE MATERIAIS A ELEMENTOS.....	41
3.5.4. UNIDADES GEOMÉTRICAS.....	44
3.5.5. DIMENSÕES GEOMÉTRICAS	47
3.6. MODELOS ESTRUTURAL E ARQUITETÓNICO PARA IFC 2x3 (AUTODESK REVIT).....	57
3.7. MODELOS ARQUITETÓNICOS PARA IFC 2x3	58
3.7.1. IDENTIFICAÇÃO DE ELEMENTOS ESTRUTURAIS.....	59
3.7.2. UNIDADES.....	59
3.7.3. DIMENSÕES GEOMÉTRICAS E MATERIAIS.....	59
3.7.4. ELEMENTOS ADICIONAIS A ESTUDAR.....	61
4 PROGRAMAÇÃO E WEB SERVICE	63
4.1. INTRODUÇÃO	63
4.2. PROGRAMAÇÃO.....	64
4.3. WEB SERVICE.....	71
5 CONCLUSÕES E SUGESTÕES.....	77
5.1. CONCLUSÕES.....	77
5.2. SUGESTÕES	80
BIBLIOGRAFIA	81
ANEXOS	89

ÍNDICE DE FIGURAS

Figura 2.1 - Fluxo tradicional da circulação de documentos. Adaptado de [16].	5
Figura 2.2 - Perda de valor na fragmentação do processo [5], adaptado de [8].	6
Figura 2.3 – BIM nos processos de construção. Adaptado de [13].	8
Figura 2.4 - Princípios básicos da ação/ interação entre humanos e BIM. Adaptado de [6], [14].	8
Figura 2.5 - Esforço, Impacto e Custo da mudança no tempo. Adaptado de [15].	9
Figura 2.6 - Fluxo de circulação de documentos: BIM. Adaptado de [16].	10
Figura 2.7 - Evolução da nomenclatura BIM como Building Information Model. Adaptado de [4].	10
Figura 2.8 - Dimensões BIM. Adaptado de [4], [6].	12
Figura 2.9 – Aproximação ao modelo real através de um modelo virtual [26].	13
Figura 2.10 - Detecção automática de erros [27].	13
Figura 2.11 – Principais benefícios do BIM para as empresas. Adaptado de [36].	15
Figura 2.12 – Possibilidade de Retorno do Investimento em função da aposta no BIM. Adaptado de [36].	16
Figura 2.13 - BIM no mundo (alguns países). Adaptado de [37].	16
Figura 2.14 - Faseamento para a implementação do BIM. Adaptado de [38].	17
Figura 2.15 - Projetos BIM: a) Flame Towers, Azerbaijão [44]; b) Terminal Oeste de Kowloon, Hong Kong [45]; c) Estação subterrânea London Victoria, Inglaterra [46]; d) Canal de Panamá, Panamá [47].	18
Figura 2.16 - Processos para estimação de custos. Adaptado de [57].	20
Figura 2.17 - IFC no conceito de OpenBIM. Adaptado de [61].	22
Figura 2.18 - História do IFC. Adaptado de [65].	23
Figura 2.19 - Estrutura do IFC [69]. Adaptado de [70].	24
Figura 2.20 – Processo de funcionamento de um Web service. Adaptado de [79].	26
Figura 2.21 - Exemplo de uma pesquisa no Google e resposta HTML. Adaptado de [80].	26
Figura 3.1 - Diferentes abordagens para a interseção: Pilar/ Viga.	30
Figura 3.2 - Estrutura de um Ficheiro do tipo "STEP-FILE" [84].	31
Figura 3.3 – Indicadores únicos, entidades e atributos. Adaptado de [84].	32
Figura 3.4 – Opções base: a) Autodesk Revit; b) Tekla Structures; c) Graphisoft ArchiCAD.	34
Figura 3.5 - Evolução do ISM. [86].	35
Figura 3.6 – Aplicabilidade da ISM [87].	36
Figura 3.7 – Modelo Analítico IFC 4 [68].	37
Figura 3.8 – Autodesk Revit: a) Modelo 3D; b) Modelo Analítico. Tekla Structures: c) Modelo 3D; b) Modelo Analítico (IFC 2x3).	38
Figura 3.9 – Identificação do Material: "IfcMaterial".	38
Figura 3.10 – "IfcColumn"; "IfcBeam"; "IfcSlab"; "IfcFooting".	40
Figura 3.11 – "IfcColumnType"; "IfcBeamType"; "IfcSlabType"; "IfcFootingType".	40
Figura 3.12 – Relação estabelecida por "IfcRelAssociatesMaterial".	41
Figura 3.13 - "IfcRelAssociatesMaterial" (IFC 4 e IFC 2x3).	41
Figura 3.14 – Relações entre "IfcRelAssociatesMaterial" e "IfcMaterial" (IFC 2x3).	42
Figura 3.15 - Relação entre "IfcRelAssociatesMaterial" e "IfcMaterial" para Laje (casos de estudo).	43

Figura 3.16 – Relações entre “IfcRelAssociatesMaterial” e “IfcMaterial” (IFC 4).....	43
Figura 3.17 – Modificação das Unidades no Modelo: Autodesk Revit e Tekla Structures (IFC 2x3). ..	44
Figura 3.18 - “IfcSIUnit”.	45
Figura 3.19 - “IfcSIUnit”: Diferentes tipos de unidades de medição.....	46
Figura 3.20 - Modelo em decímetros: a) Autodesk Revit; b) Solibri.....	47
Figura 3.21 - Autodesk Revit: Exemplo aplicado a centímetros.....	47
Figura 3.22 – Relações para obter as propriedades geométricas dos Pilares para os modelos: Autodesk Revit (IFC 2x3) e Concebido Manualmente (IFC 4).	48
Figura 3.23 – Entidades que contêm as propriedades geométricas: “IfcExtrudedAreaSolid” e “IfcRectangleProfileDef” (IFC 4 e IFC 2x3).....	49
Figura 3.24 – Relações para obter as propriedades geométricas das Vigas para os modelos: Autodesk Revit (IFC 2x3) e Concebido Manualmente (IFC 4).	50
Figura 3.25 - Relações para obter as propriedades geométricas da Laje para os modelos: Autodesk Revit (IFC 2x3) e Concebido Manualmente (IFC 4).	51
Figura 3.26 - Relações para obter as propriedades geométricas das Sapatas para os modelos: Autodesk Revit (IFC 2x3) e Concebido Manualmente (IFC 4).	51
Figura 3.27 - Determinação: Profundidade, Área e Volume - Pilares (Tekla Structures IFC 2x3).....	53
Figura 3.28 - Determinação: Profundidade, Área e Volume - Vigas (Tekla Structures IFC 2x3).....	53
Figura 3.29 - Determinação: Profundidade, Área e Volume - Laje (Tekla Structures IFC 2x3).....	53
Figura 3.30 -Entidades que contêm os valores da Profundidade, Área e Volume.	54
Figura 3.31 - Determinação: Profundidade e Área - Sapatas (Tekla Structures IFC 2x3).....	54
Figura 3.32 - Volume das Sapatas (Tekla Structures IFC 2x3).....	56
Figura 3.33 - Construção da superfície do elemento a partir das entidades "IfcPolyLoop" e "IfcCartesianPoint".	56
Figura 3.34 - Autodesk Revit: Modelo Arquitetónico.	57
Figura 3.35 – Graphisoft ArchiCAD: Modelo Arquitetónico.	58
Figura 3.36 – Trajeto genérico para obtenção das propriedades dos elementos estruturais (Autodesk Revit vs Archicad).....	59
Figura 3.37 - Determinação: Profundidade, Área - Laje (ArchiCAD IFC 2x3).....	60
Figura 3.38 - Forma da laje (ArchiCAD IFC 2x3).	60
Figura 3.39 - Cálculo da área pelo Algoritmo de Shoelace.....	61
Figura 3.40 - Representação no modelo IFC de elementos adicionais a investigar [91].....	62
Figura 4.1 – Programação, Determinação de Quantidades e Web service.....	63
Figura 4.2 - Comunicação através de uma API para IFC.	64
Figura 4.3 - Fluxograma Simplificado.....	66
Figura 4.4 - Inserir o path do ficheiro IFC.....	66
Figura 4.5 - Estrutura tipo de um dicionário em Python.....	67
Figura 4.6 - Bases de dados criadas a partir dos ficheiros IFC.	67
Figura 4.7 - Exemplo genérico da conversão da informação IFC para o dicionário “aranged_IFC”. ..	67
Figura 4.8 - Chave: "model.program()".	68
Figura 4.9 - Chave: "model.columns_properties()".	69
Figura 4.10 - Código: "model.columns_quantities ()".	69
Figura 4.11 - Código: "model_material_quantities ()".	70

Figura 4.12 - Funcionamento do Web service para IFC.	71
Figura 4.13 – Dicionários que compõem o dicionário global.	72
Figura 4.14 – Comandos em Python para a conversão da informação do dicionário global para XML.	73
Figura 4.15 – Funções “WEBSERVICE” e “FILTERXML” do Microsoft Excel 2013.....	73
Figura 4.16 - Exemplo de utilização da função “WEBSERVICE”.	73
Figura 4.17 - Exemplo da aplicação da função “FILTERXML” do Microsoft Excel 2013.....	74
Figura 4.18 - Obtenção do Código XML para importação global da informação no Microsoft Excel 2013.....	74
Figura 4.19 - Opção "From XML Data Import" do Microsoft Excel.	75

ÍNDICE DE EQUAÇÕES

Fórmula 3.1 – Algoritmo de <i>Shoelace</i>	61
--	----

ÍNDICE DE QUADROS

Quadro 3.1- Análises comparativas entre diferentes modelos e versões IFC.....	33
Quadro 3.2 - Softwares: Modelos Estruturais e Arquitetónicos.	35
Quadro 3.3 - Identificação de elementos estruturais (Modelos Estruturais).	40
Quadro 3.4 – Trajeto para a obtenção das propriedades geométricas de Pilares.	49
Quadro 3.5 – Comparação entre os trajeto das vigas, lajes e sapatas e pilares para IFC 2x3.	52
Quadro 3.6 - Elementos adicionais passíveis de ser incluídos neste estudo [91].	62
Quadro 4.1 - Comandos para obtenção dos resultados finais (<i>Python</i>).	68
Quadro 4.2 - Unidades e Arredondamentos a adotar. [52].....	70

SÍMBOLOS E ABREVIATURAS

2D – DUAS DIMENSÕES

3D – TRÊS DIMENSÕES

4D – QUATRO DIMENSÕES

5D – CINCO DIMENSÕES

6D – SEIS DIMENSÕES

AEC – ARQUITETURA, ENGENHARIA E CONSTRUÇÃO

AEC-FM – ARCHITECTURE, ENGINEERING, CONSTRUCTION AND FACILITY MANAGEMENT

AGC – ASSOCIATED GENERAL CONTRACTORS OF AMERICA

API – APPLICATION PROGRAMMING INTERFACE

ASCII - AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE

BDA – BUILDING DESIGN ADVISOR

BDS – BUILDING DESCRIPTION SYSTEM

BIM – BUILDING INFORMATION MODEL/MODELING

BPM – BUILDING PRODUCT MODELS

CAD – COMPUTER-AIDED DESIGN

CAEDADS – COMPUTER-AIDED ENGINEERING AND ARCHITECTURAL DESIGN SYSTEM

CATIA – COMPUTER-AIDED THREE-DIMENSIONAL INTERACTIVE APPLICATION

CHEOPS – HIGHER ORDER LOGIC-SUPPORTED DESIGN FOR COMPLEX DATA-PROCESSING SYSTEMS

CPP – CÓDIGO DOS CONTRATOS PÚBLICOS

EAU – EMIRATOS ÁRABES UNIDOS

ERP – ENTERPRISE RESOURCE PLANNING

EUA- ESTADOS UNIDOS DA AMÉRICA

FEUP – FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

FM – FACILITY MANAGEMENT

GDS – GLOBAL DISTRIBUTION SYSTEM

GLIDE – GRAPHICAL LANGUAGE FOR INTERACTIVE DESIGN

GUID – GLOBAL UNIQUE IDENTIFIER

HTML - HYPERTEXT MARKUP LANGUAGE

HTTP – HYPERTEXT MARKUP PROTOCOL

IAI – INTERNATIONAL ALLIANCE FOR INTEROPERABILITY

IFC – INDUSTRY FOUNDATION CLASSES

IPD – INTEGRATED PROJECT DELIVERY
INESC – INSTITUTO DE ENGENHARIA DE SISTEMAS E COMPUTADORES DO PORTO
ISM – INTEGRATED STRUCTURAL MODELING
LNEC – LABORATÓRIO NACIONAL DE ENGENHARIA CIVIL
MIEC – MESTRADO INTEGRADO ENGENHARIA CIVIL
MIME – MULTIPURPOSE INTERNET MAIL EXTENSIONS
MTQ – MAPA DE TRABALHOS E QUANTIDADES
PIM – PRODUCT INFORMATION MODELS
PRONIC – PROTOCOLO PARA A NORMALIZAÇÃO DA INFORMAÇÃO TÉCNICA NA CONSTRUÇÃO
REST – REPRESENTATIONAL STATE TRANSFER
RFID – RADIO-FREQUENCY IDENTIFICATION
RUCAPS – REALLY UNIVERSAL COMPUTED AIDED PRODUCTION SYSTEM
STEP – STANDARD FOR THE EXCHANGE OF PRODUCT MODEL DATA
TEI – TOTAL ENTERPRISE INTEGRATION
TIC – TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO
URI – UNIFORM RESOURCE IDENTIFIER
URL – UNIFORM RESOURCE LOCATOR
W3C – WORLD WIDE WEB CONSORTIUM
XML – EXTENSIBLE MARKUP LANGUAGE

1

INTRODUÇÃO

1.1. ENQUADRAMENTO

A evolução do Homem e as suas descobertas têm marcado a história da Humanidade em todos os grandes setores. Relativamente ao setor da construção, existem manifestações de Engenharia Civil um pouco por todo o mundo, estando a sua grandiosidade, pela forma e dimensão das estruturas edificadas, presente em inúmeros testemunhos deixados pelos Egípcios, Gregos, Romanos, entre outros.

Mesmo alguns dos materiais empregues na atualidade são oriundos destas épocas, existindo, inclusive, indícios físicos da aplicação de, por exemplo, betões e materiais de construção em grandes obras de arte, como é o caso do Coliseu de Roma e do Panteão [1].

Algumas destas obras de arte suscitam, ainda, na atualidade, curiosidade pelo modo como foram conseguidas numa época em que os meios tecnológicos tinham um campo de atuação bastante limitado. Não obstante a este argumento, o Homem ditou o seu próprio destino e alcançou obras de arte cuja explicação física é, ainda hoje, uma incógnita.

É exemplo disso a cúpula da catedral de *Santa Maria del Fiore* em Florença, Itália, da autoria do Arquitecto *Filippo Brunelleschi*. Este monumento tem uma altura de cerca de 114 metros e a sua construção foi levada a cabo no início do Renascimento, há quase seis séculos atrás. A sua cúpula é única por vários motivos, destacando-se a não utilização de um sistema central de suporte aquando da sua construção e, por isso, a engenhosidade necessária para a estrutura se suportar por si só, sendo os métodos empregados na sua construção ainda difíceis de compreender [2].

Evidentemente, a Engenharia Civil na Antiguidade era mais uma arte do que uma própria ciência, talvez pelas dificuldades em adequar o conhecimento existente à própria Engenharia. No entanto, com o decorrer dos tempos, devido às necessidades do Homem em se adaptar a diferentes cenários, não só foi possível a evolução de todo o seu legado, como também da sua aplicabilidade prática [3].

Atualmente, em plena era tecnológica, onde as tecnologias da informação são introduzidas a uma velocidade estonteante em vários domínios, também no setor da construção estas começam a ser mais evidentes e com mais profissionais suscetíveis à sua implementação e à consequente mudança de paradigma.

Pelo âmbito deste trabalho, destaca-se o BIM (*Building Information Model* ou *Building Information Modeling*) que surge não só como o fio condutor entre a inovação e a otimização de processos construtivos, mas também como um desafio ao próprio engenho do Homem. Por sua vez, a otimização de processos construtivos conduz a menores tempos de duração de tarefas e a um melhor aproveitamento de recursos e, por isso, a custos globais inferiores. O Homem, apesar de ser um ser obstinado a

aperfeiçoar as suas virtudes, é também um ser exigente que apenas aceita a mudança mediante provas irrefutáveis das suas vantagens. Por esta razão, todo processo de evolução torna-se um pouco moroso. No presente, o BIM já apresentou indícios que poderá ser uma mais-valia na indústria de Arquitetura, Engenharia, Construção e Gestão de Edifícios e, cada vez mais, adquire novos seguidores, estando até já em fase de implementação à escala interna de alguns países.

A partilha de informações de modelos BIM, com os diversos intervenientes no processo construtivo, é possível através de um formato aberto denominado IFC (*Industry Foundation Classes*). Este formato, sendo gratuito, ainda não possibilita um nível de interoperabilidade suficiente que garanta a não existência de lacunas entre os modelos transferidos através de ferramentas computacionais. No entanto, as falhas são cada vez menores e, por isso, a sua aplicabilidade cada vez superior.

Como a conversão de modelos BIM para, o formato aberto, IFC é conseguida através da acoplação de toda a informação numa base de dados do tipo texto, então, geram-se oportunidades de investigação concernentes ao modo como a informação é reunida e estruturada. Com efeito, podem-se conceber algoritmos para a manipulação desta base de dados e incorpora-los num *Web service*. Desta forma, suprime-se a necessidade de instalação de ferramentas específicas em computadores pessoais.

1.2. ÂMBITO E OBJETIVOS

A presente dissertação insere-se no ramo de especialização de Construções Civis do Mestrado Integrado Engenharia Civil (MIEC) da Faculdade de Engenharia da Universidade do Porto (FEUP). Pretende-se com este documento, apresentar um estudo relativo ao formato aberto de troca de dados entre ferramentas BIM, o IFC, estabelecendo metodologias padrão para a quantificação de trabalhos relativos a estruturas. Não faria sentido não iniciar o estudo nesta área, uma vez que as informações de qualquer obra são numa primeira fase limitadas e aumentam gradualmente até atingirem o nível de detalhe exigido. Com efeito, pretende-se, o desenvolvimento de um modelo programático associado às metodologias referidas, de modo a automatizar todo o processo. Por sua vez, os serviços disponibilizados por esta ferramenta poderão ser acessíveis via internet através de um navegador, recorrendo, para tal, a um *Web service* como meio de comunicação entre um programa e a internet.

Muitas das ferramentas BIM disponíveis no mercado apresentam capacidades que permitem a determinação de quantidades de materiais, utilizando para tal o seu formato interno. A ferramenta idealizada distingue-se destes programas por permitir o acesso gratuito e por servir-se das bases de dados incorporadas no modelo IFC.

É apresentado neste trabalho um conjunto de ferramentas informáticas originais que visam (1) a extração automática de quantidades de trabalhos (de estruturas) dado um modelo BIM em formato aberto IFC, bem como (2) a submissão de modelos IFC para tratamento remoto, usando *Web services*. A vantagem da utilização de *Web services* passa essencialmente pela não necessidade de recorrer à instalação de programas internamente nos computadores, utilizando a internet como ferramenta neste processo, e pela possibilidade de partilha de serviços para outras aplicações.

1.3. ESTRUTURA DA DISSERTAÇÃO

No que concerne à estrutura/ organização desta dissertação, esta é dividida em cinco (5) capítulos distintos:

- Capítulo 1 – “Introdução”, estabelece-se o enquadramento ao tema abordado numa perspetiva histórica, desde os primórdios até à atualidade. Seguidamente, são apresentados os objetivos traçados para este trabalho, assim como a organização deste documento.
- Capítulo 2 – “Estado da Arte”, como o nome indica, é feito um enquadramento aos temas abordados ao longo da presente dissertação nomeadamente, o estado do setor da Construção Civil, o BIM como auxílio neste processo, o formato aberto de partilha de modelos BIM: o IFC, as regras de medição aplicáveis ao território nacional, a correlação entre estas regras e o IFC e uma introdução aos *Web services*.
- Capítulo 3 – “Extração de Quantidades Suportada por Modelos BIM”, são concebidos modelos iguais de uma mesma estrutura tipo a partir de diferentes ferramentas BIM com o intuito de contabilizar trajetos padrão e casos pontuais, a partir dos modelos IFC gerados.
- Capítulo 4 – “Programação e *Web service*”, com a informação obtida a partir do capítulo anterior, produz-se um modelo programático em *Python* que possibilite o cálculo automático de quantidades. Posteriormente, transpõe-se este serviço para a internet, recorrendo, para tal, a um *Web service*.
- Capítulo 5 – “Conclusões e Sugestões”, apresentam-se as ilações obtidas com este estudo e sugestões para trabalhos futuros.

2

ESTADO DA ARTE

2.1. INTRODUÇÃO

No presente capítulo, apresenta-se uma introdução aos temas de interesse abordados ao longo da dissertação, tendo em vista uma primeira abordagem ao objetivo primário deste documento: a obtenção de medições de estruturas a partir do IFC, conseqüente construção de um modelo programático para a sua determinação automática e, finalmente, a sua implementação a uma escala global.

A escolha das áreas abordadas neste capítulo refletem esse mesmo objetivo e são: problemas na indústria da Arquitetura, Engenharia, Construção (AEC); apresentação do conceito, história, importância e estado de implementação do BIM; IFC como formato livre de partilha de modelos BIM; regras de medição aplicáveis ao território nacional; BIM no contexto de medições; *Web services* para a implementação de um programa num *Website*.

A importância deste capítulo reflete-se pela exposição de conceitos base que permitam ao leitor um enquadramento geral ao tema e pela apresentação de referências a trabalhos desenvolvidos neste âmbito, com um carácter crítico e reflexivo.

2.2. PROBLEMAS E SOLUÇÕES NA INDÚSTRIA AEC

Atualmente, a indústria da Arquitetura, Engenharia e Construção ainda depende, em grande parte, de documentos em papel para a comunicação entre os diversos intervenientes no processo construtivo, o que origina a fragmentação de processos [4].

Pela Figura 2.1, constata-se o nível de complexidade que o conjunto das trocas de informações pode assumir, assemelhando-se este a uma “teia”. O recurso a processos tradicionais obriga à existência de um grau de interdependência elevado entre o trabalho desenvolvido por profissionais nas diferentes fases de uma obra.

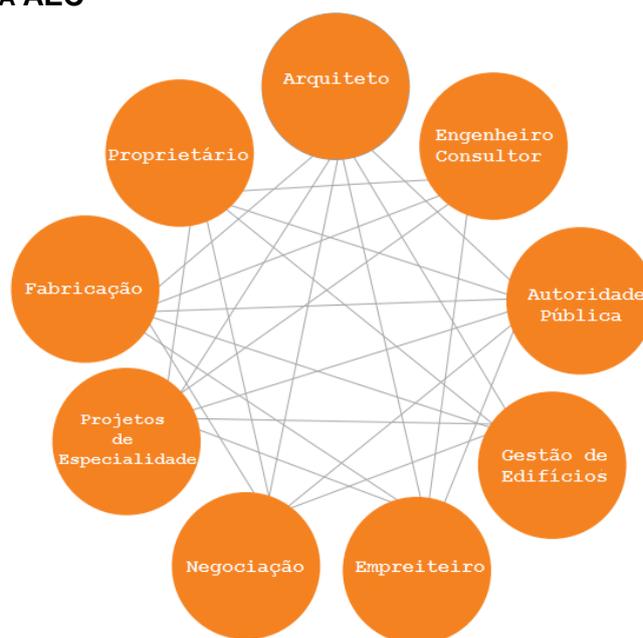


Figura 2.1 - Fluxo tradicional da circulação de documentos. Adaptado de [16].

Essencialmente, as fases dividem-se em Projeto, *Procurement*, Construção e Utilização, sendo que, na transposição de uma fase para a outra, as informações poderão não ser geridas da melhor forma, resultando na sua perda parcial [5]. Existe, portanto, um problema na eficácia da gestão da informação, causado pela interoperabilidade entre sistemas da informação.

Os motivos que estão na base destas perdas de informação são [6]–[8]:

- Circulação de diferentes versões de documentos que, por sua vez, podem conduzir a erros de projeto e conseqüentemente a atrasos e derrapagens nos custos;
- Dificuldade de visualização global, compreensão e conseqüente comunicação da informação;
- A generalidade da documentação processar-se segundo um formato 2D, sendo necessária a sua adaptação para o contexto real da obra, ou seja, para uma dimensão 3D;
- Presença de indivíduos com diferentes naturezas e culturas, o que gera dificuldade na transmissão de ideias;
- Introdução repetida de dados;
- Não aproveitamento de informações adquiridas em projetos antecedentes.

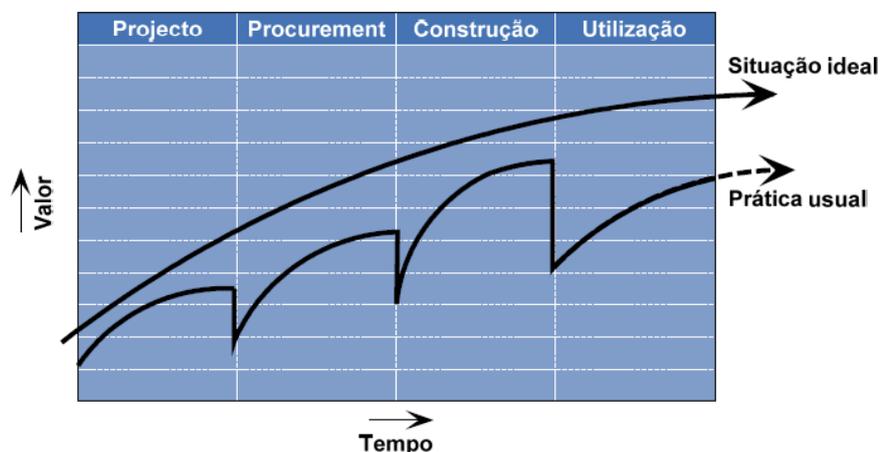


Figura 2.2 - Perda de valor na fragmentação do processo [5], adaptado de [8].

Neste setor, como em qualquer outro, subsiste uma procura crescente, no sentido de melhorar a performance e a produtividade [9]. Neste sentido, e com o propósito de confrontar a dificuldade referida de comunicação, foram adotadas diferentes estratégias no âmbito das Tecnologias da Informação e Comunicação (TIC), nas quais se destaca, pelo sucesso obtido, o *Enterprise Resource Planning* (ERP) [7]. Com a sua implementação, empresas com retornos de investimento na ordem dos 25% - 50%, podem ampliá-los para valores compreendidos entre 50% e 100% (ou até superiores). Contudo, os valores esperados situam-se usualmente entre os 25% e 100% [10].

Technology, including TEI¹, can succeed only if the people involved want it to succeed.

Gary A. Langenwalter

¹ TEI (*Total Enterprise Integration*), supra-conjunto do ERP, representa os meios de comunicação de uma empresa com trabalhadores, clientes e fornecedores. Os TEI's permitem esclarecer dúvidas e projetar melhor os resultados que cada decisão acarreta.

Todavia, os problemas existentes não se traduzem apenas devido a falhas nos processos comunicativos, mas, também, através de rivalidades entre membros de uma mesma equipa (os quais lutam primeiro pelos objetivos pessoais em detrimento dos objetivos do coletivo), da transferência do risco entre estes e dos litígios [6].

A temática dos litígios ou conflitos na construção é, hoje-em-dia, muito corrente neste setor e envolve diversas entidades, nomeadamente Dono de Obra, Projetistas, Empreiteiros e entidades de Fiscalização. Os conflitos incidem essencialmente em questões relacionadas com medições, prazos de execução, irregularidades de projeto e segurança, desperdícios e avarias e defeitos, verificando-se que a apuramento de responsabilidades é dificultada pela tentativa das partes envolvidas em ilibarem-se destas mesmas [11]. Estas situações afetam a qualidade do setor da construção e podem resultar na perda de confiança pública, em insatisfações e em desconsideração [5].

Apesar do sucesso de implementação do tipo de estratégias atrás referidas, os problemas neste setor continuam presentes, sendo o BIM (*Building Information Model*) a tecnologia de topo que pretende minimizar não só os problemas comunicativos, mas também todos aqueles apresentados através da reformulação das práticas de trabalho. O custo associado às ferramentas BIM apresenta um grau de variação elevado (por exemplo, 4.000 e 15.000€) e, por isso, são importantes estudos de mercado que permitam perceber quais as necessidades da empresa e as ofertas reais [12].

Qualquer solução a adotar tem que se apoiar numa coordenação global rigorosa, de modo a reduzir a acumulação de erros e, conseqüentemente permitir o aumento do benefício geral.

2.3. BIM

2.3.1. CONCEITO

Atualmente, existe uma tendência de indivíduos exteriores ao BIM, pensarem neste como um programa ou *software*, o que é incorreto. O BIM pode ser interpretado como *Building Information Modeling* ou *Building Information Model*. Estes conceitos, apesar de semelhantes, são distintos. Enquanto o primeiro se refere a um processo de introdução e partilha de informação entre os demais parceiros para a melhoria de procedimentos (processo colaborativo), o segundo relaciona-se com um objeto ou modelo contendo todas as informações relevantes de um edifício [4]. Por outras palavras, e mais simplificada, enquanto o primeiro define um processo contínuo de trabalho, o segundo refere-se a um produto final, concebido à imagem de um modelo virtual de edifício inteligente.

As informações contidas no modelo podem assumir uma dimensão 2D, 3D, 4D (planeamento de tarefas), 5D (estimação de custos), 6D (gestão de edifícios) ou nD (como por exemplo, informações relativas à segurança e à sustentabilidade) [6].

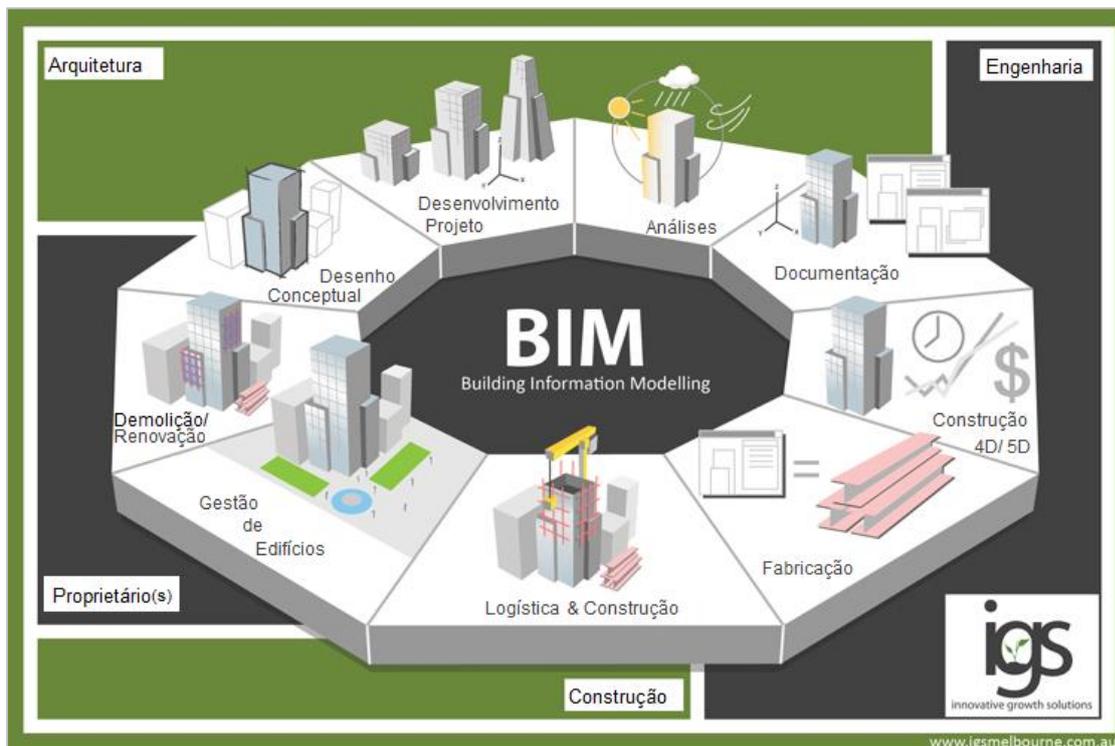


Figura 2.3 – BIM nos processos de construção. Adaptado de [13].

O BIM é uma das mais importantes e promissoras inovações da atualidade para os setores da Arquitetura, Engenharia, Construção e Gestão de Edifícios (AEC-FM). Através deste, é possível a criação de modelos virtuais precisos de um edifício, de apoio a todas as fases de uma obra, permitindo, desta forma, uma melhor análise e controle do que aquela verificada por processos manuais [4]. Pela Figura 2.3, evidencia-se que o BIM, não só está presente em todo o conjunto de processos referentes ao período de vida útil de um edifício, desde a sua conceção até à sua demolição, mas também em todos os processos alusivos ao projeto, estando, por isso, presente em todo o seu ciclo de vida.

Os princípios básicos do BIM são também os princípios básicos da ação e interação entre humanos: visualização; compreensão; comunicação; colaboração. Cada um destes conceitos reforça os outros três [6].

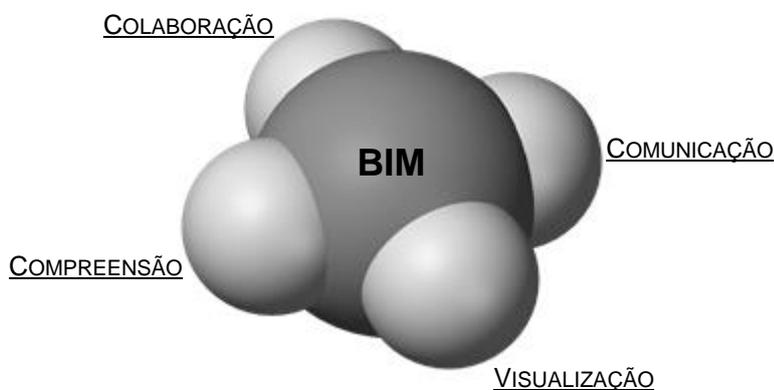


Figura 2.4 - Princípios básicos da ação/ interação entre humanos e BIM. Adaptado de [6], [14].

“BIM is not a single building model or a single database”; “BIM is not a replacement for people”; “BIM is not perfect”; “BIM is not Revit (or ArchiCad, or Bentley)”; “BIM is not just 3D”; “BIM does not have to be 3D”; “BIM is not complete”.

“BIM is an information based system that builds long-term value innovation.”

Finith Jernigan

Com o desenvolvimento tecnológico surgem maiores exigências ao nível de projeto para o tornar mais eficiente e com o menor custo possível, sendo, por isso, necessária uma adaptação no modo como os intervenientes pensam e agem. O BIM, como projeto colaborativo, é uma solução viável para alcançar tais objetivos. Na figura seguinte, apresenta-se um gráfico relativo ao esforço, impacto e custo da mudança de processos no tempo [15].

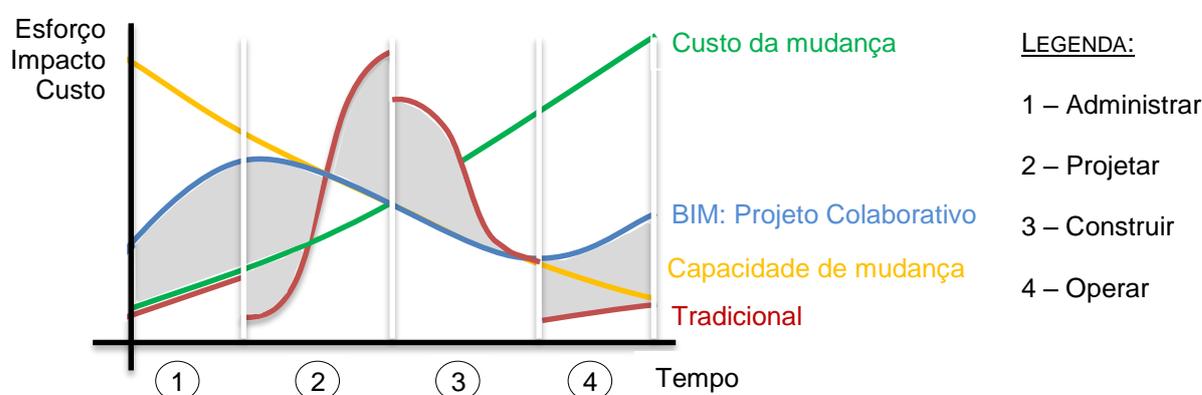


Figura 2.5 - Esforço, Impacto e Custo da mudança no tempo. Adaptado de [15].

Num projeto colaborativo, as metas de todos os intervenientes no projeto devem convergir para um mesmo objetivo primário, o sucesso da sua equipa de trabalho e, em simultâneo, não interferir com os do(s) proprietário(s). O BIM, utilizado como ferramenta neste processo, pode mostrar-se uma mais-valia para alcançar os objetivos traçados pela equipa, uma vez que torna todo o processo mais transparente, através da exposição de todas as debilidades do projeto [6]. Enquanto através dos processos tradicionais ocorre uma perda de informação na transição entre as várias fases (Figura 2.2 e 2.5, a vermelho), com um processo colaborativo (Figura 2.5 **Error! Reference source not found.**, a azul) tal não ocorre. É evidente a existência de obstáculos à mudança tecnológica e de práticas de trabalho, não só pelo custo associado, mas também pela suscetibilidade dos profissionais à própria mudança. Além disso, esta dificuldade é intensificada pela heterogeneidade de todas as variáveis existentes no sector da construção [5].

Em oposição à Figura 2.1, na Figura 2.6, verifica-se uma mudança no modo como a informação circula, uma vez que, através do BIM, é criada uma fonte central de informação [16]. Por sua vez, esta centralização permite que todos os intervenientes no processo construtivo trabalhem com a última versão do modelo, eliminando a problemática da circulação de diferentes versões do projeto.

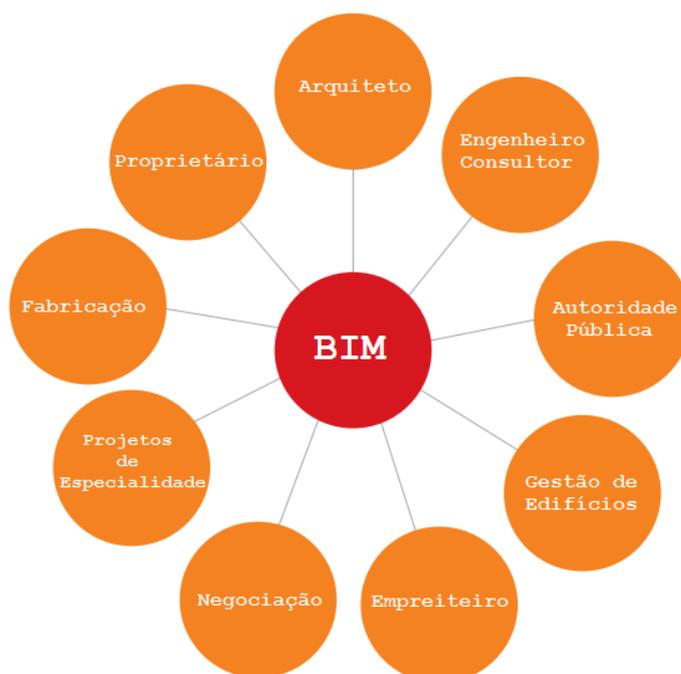


Figura 2.6 - Fluxo de circulação de documentos: BIM. Adaptado de [16].

2.3.2. BREVE HISTÓRIA

O conceito inerente ao BIM (*Building Information Model*) nasce por iniciativa de *Charles M. “Chuck” Eastman*, atual professor e diretor do “*Digital Building Laboratory*”, sendo este apresentado, nas suas obras, como *Building Product Model*. O termo atual do BIM, como *Building Information Model* nasce por iniciativa de *Van Nederveen e Tolman*, em 1992, no artigo: [17].

Segundo *Charles M. “Chuck” Eastman*, ao modelo de edifício virtual atribuíram-se as designações de “*Building Product Models*” (BPM), nos Estados Unidos da América, e de “*Product Information Models*” (PIM), na Europa. Desta forma, eliminando o termo “*Product*” e unindo estas duas denominações surge a designação “*Building Information Model*” (BIM), nomenclatura atualmente empregue [4].

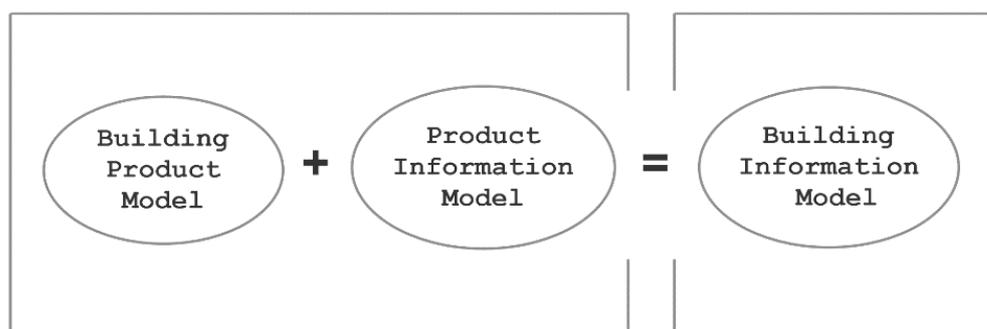


Figura 2.7 - Evolução da nomenclatura BIM como *Building Information Model*. Adaptado de [4].

O termo *Building Information Modeling*, como um processo contínuo de trabalho, surge posteriormente por iniciativa de *Phil Bernstein*, arquiteto e coordenador na *Autodesk* [15], sendo popularizado com o apoio de *Jerry Laiserin* [17].

Apesar do seu impacto ter-se revelado maior nos últimos anos, com uma maior preferência, por grandes empresas, em detrimento das tecnologias CAD (*Computer Aided-Design*), tendo até, sido deliberada, em Inglaterra, a obrigatoriedade de implementação do BIM em todos os projetos públicos para o ano de 2016, algumas das ideias inerentes ao BIM surgiram há cerca de cinco (5) décadas, nos anos 60 [18].

Esta temática foi iniciada no artigo “*Augmenting Human Intellect*”, em 1962, onde é apresentada uma visão futura da arquitetura por *Douglas C. Englebart*, conhecido pela criação do “rato” de computador [19]. Neste artigo, são já introduzidas as ideias de modelação orientada por objetos, manipulação paramétrica e banco de dados relacional [20].

Mais tarde, na década de 70, surgem publicações no “*AIA Journal*” (*Journal of the American Institute of Architects*) e, posteriormente, no “*Carnegie-Mellon University*”, por *Charles M. “Chuck” Eastman*, onde é apresentado o protótipo “*Building Description System*” (BDS), ou em português, Sistema de Descrição de Edifícios [4]. Este *software* foi o primeiro a descrever a adição e a recolha de informações de cada elemento singularmente, sendo estimado que a sua utilização poderia reduzir os custos de construção em cerca de cinquenta por cento (50%). Contudo, subsiste a dúvida se este foi realmente empregue em algum projeto real, uma vez que surgiu numa época pouco desenvolvida em termos computacionais [20]. Posteriormente, em 1977, surge o projeto GLIDE (*Graphical Language for Interactive Design*), também por sua iniciativa, onde são apresentadas noções de geração automática de medições e de estimativa de custos, interdependência de desenhos e base de dados relacional do modelo, ou seja, atributos mais específicos e atuais do BIM [21].

Apresenta-se, de seguida, uma lista exaustiva de ferramentas concebidas nesta área: *Accugraph systems*; *ARCH-model*; *CAEADS*; *CATIA*; *Cedar*; *CHEOPS*; *EdCADD*; *Formative Technology*; *GE/CALMA*; *HARNESS*; *OXSYS* mais tarde subdividido em *BDA (Building Design Advisor)* e *GDS*; *Pro/Engineer*; *Radar CH* atualmente conhecido como *ArchiCAD Reflex*; *RUCAPS (Really Universal Computer Aided Production System)*, mais tarde substituído pelo *software Sonata*, o qual é precursor do *Autodesk Revit*; *Scribe*; *SKOK* (mais tarde *DrawBase*) *system*; *SSHA*; *Tekla*; entre outras [4], [20]. Destaca-se o *software Scribe* como sendo o primeiro *software* BIM a poder ser utilizado num computador pessoal, apesar da generalidade da documentação existente indicar que este registo pertence ao *Radar CH (ArchiCAD)* [22]. Acrescenta-se que a maioria das ferramentas computacionais referidas foram desenvolvidas maioritariamente no final e no início das décadas de 1970 e 1980, respetivamente [23].

Atualmente, existem *softwares* BIM para os mais diversos setores/ áreas da construção como, por exemplo:

- a. Arquitetura (*Autodesk Revit*, *Vectorworks Architect*, *Graphisoft ArchiCAD*, *Bentley Architecture*);
- b. Sustentabilidade (*Autodesk Ecotect Analysis*, *Autodesk Green Building Studio*, *Graphisoft EcoDesigner*, *Bentley Tas Simulator* e *Hevacomp*);
- c. Estruturas (*Autodesk Revit*, *Tekla Structures*, *CypeCAD*, *Bentley RAM*, *STAAD*, *ProSteel E Structural Modeler*, *Autodesk Robot Structural Analysis*);
- d. Instalações (*Autodesk Revit*, *Bentley Hevacomp Mechanical Designer*, *CADMEP (CADduct / CADmech)*, *Gehry Technologies - Digital Project MEP Systems Routing*);
- e. Construção (*Autodesk Navisworks*, *Solibri Model Checker*, *Tekla BIMsight*, *Bentley ConstrucSim*, *Innovaya*, *Bentley ConstrucSim*);
- f. Gestão de Edifícios (*FM:Systems FM:Interact*, *Bentley Facilities*, *Vintocon ArchiFM* (para o *ArchiCAD*), *Onuma System*).

2.3.3. VANTAGENS E DESVANTAGENS DO MODELO VIRTUAL

2.3.3.1. Vantagens

Como em qualquer tecnologia, o BIM apenas trará proveitos se empregado adequadamente, sendo que a utilização de ferramentas BIM não é, por si só, suficiente para alcançar o sucesso. Tal deve-se à existência de diversas condicionantes que influenciam os processos construtivos, como por exemplo, a sua duração, o seu desenvolvimento, a procura de recursos e a estabilidade financeira [24].

A primeira grande vantagem do BIM, já referida, é a sua extensibilidade a todas as fases do ciclo de vida de uma obra [4], sendo as tomadas de decisões decorrentes de todo o processo construtivo realizadas com maior segurança, uma vez que o nível de detalhe da informação e as análises construtivas são conseguidas mais rapidamente [25]. A antecipação de decisões conduz, também, a menores custos globais, uma vez que tal implica que o processo construtivo não se encontre tão avançado quanto seria de esperar através de práticas tradicionais. Contudo, para usufruir das vantagens inerentes a esta característica, deverão ser introduzidas alterações nos procedimentos de contratação de projetos, de modo a não ocorrer transferência de riscos, do Dono de Obra para o(s) projetista(s), relacionados com a não conclusão do projeto [7].

Numa primeira fase, gera-se um modelo tridimensional (3D), a partir do qual poderão obter-se, automaticamente, todos os desenhos bidimensionais (2D) pretendidos e em que qualquer modificação introduzida no modelo influenciará todos os desenhos (planta, cortes e alçados) que se encontrem interligados à referida alteração. A esta propriedade “inteligente” do BIM dá-se a designação de modelação paramétrica. Assim como os desenhos, também a documentação escrita poderá ser obtida de modo automático [4]. Posteriormente, segue-se a adição ao modelo das “n” dimensões pretendidas, exemplificadas na figura seguinte.

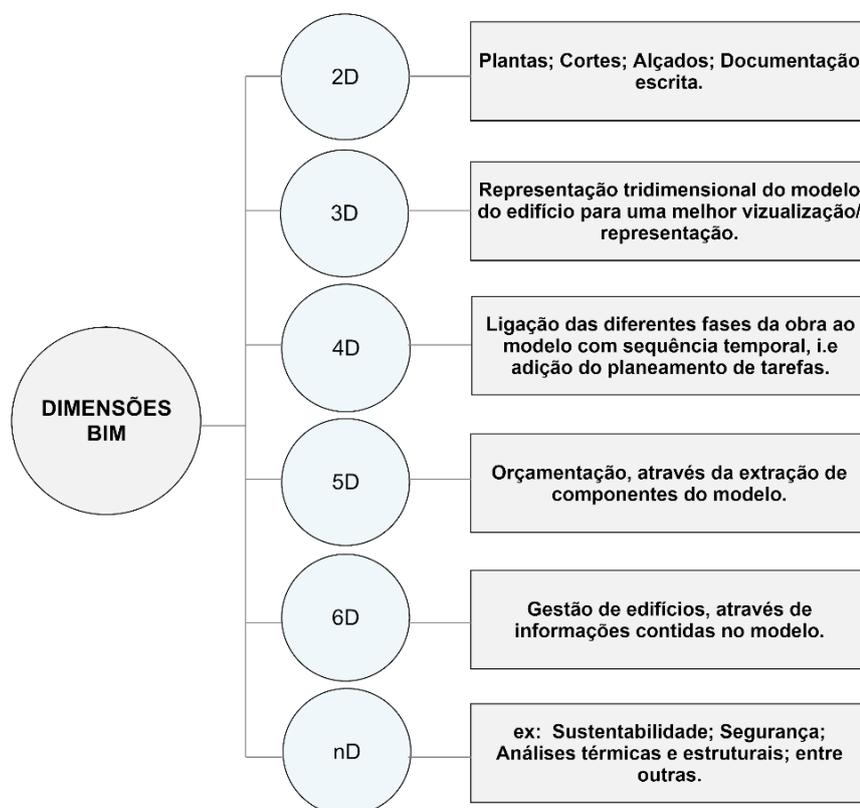


Figura 2.8 - Dimensões BIM. Adaptado de [4], [6].

A constituição de um modelo tridimensional permite uma maior aproximação à realidade, facilitando, por isso, a obtenção de um produto coerente com os requisitos inicialmente solicitados pela(s) entidade(s) contratante(s) e a comunicação entre os intervenientes no processo construtivo, reduzindo erros e custos. Na Figura 2.9, à esquerda, é apresentada uma visualização tridimensional (3D) de um modelo e, à direita, o resultado final alcançado.



Figura 2.9 – Aproximação ao modelo real através de um modelo virtual [26].

As ferramentas BIM permitem que a conversão de unidades de todos os elementos do projeto seja acessível através de um único comando. Por exemplo, tal característica permite atingir um nível de sincronização ótimo entre a equipa de projeto e fornecedores, na medida em que as unidades de trabalho adotadas por estes poderão não estar em concordância [25].

Um outro aspeto inovador do BIM consiste na deteção automática de erros no projeto, identificando os conflitos existentes antes do início dos trabalhos. Por um lado, esta característica permite atenuar a acumulação de erros e omissões de projeto e, por outro, possibilita a criação e simulação de múltiplos cenários que, por sua vez, fomentam a criatividade de cada um [4].

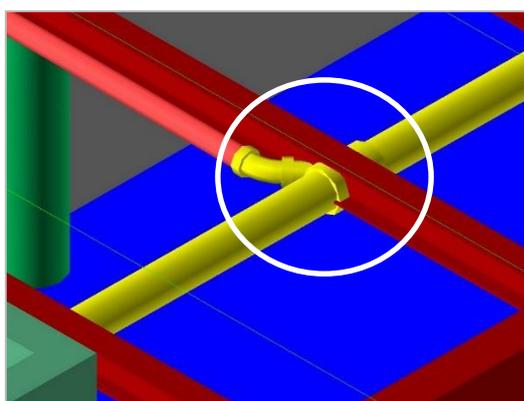


Figura 2.10 - Deteção automática de erros [27].

O modelo tridimensional (3D) pode, também, ser utilizado como uma ferramenta gráfica para ilustrar, por exemplo, planos de evacuação, falhas, progressão da obra no tempo, entre outros [28]. A simulação gráfica da construção da obra no tempo é alcançada através da adição das diferentes tarefas segundo

uma ordem cronológica, recorrendo para tal a um cronograma/ diagrama de *Gantt*. O planeamento de tarefas é uma atividade que, com o decorrer dos anos, tem vindo a adquirir cada vez mais importância, na medida em que o não cumprimento de prazos previamente estabelecidos pode resultar em multas severas. Neste sentido, o planeamento adquire uma índole fundamental, pois é necessário garantir, com uma boa estratégia prévia e cuidadosamente planificada, que os objetivos traçados sejam cumpridos.

Refere-se que a criação de um modelo de edifício virtual, contendo as informações detalhadas de cada elemento, permite que a elaboração das estimativas orçamentais se execute mais simplificada, não só em termos de precisão, mas também em tempo e trabalho, uma vez que será possível a extração automática das informações relevantes para a estimativa. Acrescenta-se que este atributo é relevante na medida em que permite simular diferentes hipóteses em tempo real durante todas as fases do processo construtivo, permitindo, por isso, um maior controle de custos [29]. Uma análise mais detalha a esta característica, justificada pelo âmbito deste documento, é apresentada mais à frente no subcapítulo 2.5.

O *Building Information Model* não é limitado a construções novas, podendo, também, ser aplicado a edificações já existentes. Tal pode ser alcançado, recorrendo a levantamentos 3D através de varrimento por laser. No entanto, pelo custo abrupto que esta tecnologia acarreta, a sua presença é ainda bastante limitada. Além de que, a sua aplicação requer um grau de afinidade elevado para com BIM, de modo a que a exportação de dados seja precisa e inequívoca [30].

Qualquer informação relevante pode ser adicionada ao modelo virtual como, por exemplo, registos referentes a encomendas, fabricação, prazos de entregas [28], localização de materiais através de informações respeitantes a etiquetas de identificação por radiofrequência, *Radio-Frequency Identification* (RFID) [31], entre outros. A existência de uma fonte central de informação permite que além de todo o trabalho referente à gestão do edifício, também, todo aquele alusivo à sua manutenção seja facilitado, uma vez que poderão ser adicionadas todas as informações pretendidas para este efeito e acedidas quando se mostrarem necessárias. Este atributo permite a conceção de um cadastro do edifício.

Todas as propriedades referidas representam algumas das vantagens mais importantes associadas ao BIM. Se utilizadas corretamente, estas permitem potenciar os resultados ao nível da comunicação entre os demais parceiros, da qualidade de produção, do aspeto estrutural, dos prazos, dos custos, da segurança e do controle e funcionamento do edifício, tendo como base um processo colaborativo e uma gestão eficiente da informação [4], [28].

2.3.3.2. Desvantagens

As desvantagens da utilização do BIM podem-se subdividir em dois grupos distintos, nomeadamente desvantagens contratuais e técnicas [28].

As questões relacionadas com direitos de propriedade da informação representam uma desvantagem, uma vez que podem gerar conflitos entre proprietários e equipa projetista [32] citado em [28]. Esta situação deve ser solucionada através da exposição dos direitos e responsabilidades nos documentos referentes ao contrato [33] citado em [28]. Além disso, podem gerar-se problemas de licenciamento nas informações incorporadas no modelo por alguém que não arquitetos, engenheiros e proprietários [34].

Uma questão pertinente consiste em como controlar a origem da informação introduzida no modelo, uma vez que, caso esta não esteja correta, poderá ser necessária a apuração de responsabilidades. Por exemplo, a utilização de modelos BIM para um fim, que não aquele para o qual foram produzidos pode conduzir à constituição de informações incorretas que podem afetar toda a fidelidade do modelo.

Com o BIM é adicionada uma nova parcela ao custo global da obra que advém do tempo despendido em adicionar e rever toda a informação do modelo. Todavia, o saldo entre a diminuição e o incremento de custos é positivo [34].

2.3.4. IMPLEMENTAÇÃO

Apesar de todos os obstáculos à implementação do BIM, este tornou-se o tema central das Tecnologias de Informação e Comunicação (TIC) no setor da Arquitetura, Engenharia e Construção (AEC), sendo, maioritariamente, empregue para os seguintes fins: Detecção automática de erros de projeto; Estimação de custos; Conceção de modelos arquitetónicos e construtivos; Coordenação das várias fases do projeto; Gestão de edifícios com o auxílio de sensores para a deteção de irregularidades; *Procurement* e sistemas de licitação integrada para modeladores paramétricos [35]. Na figura seguinte, apresenta-se uma sondagem realizada pela *McGraw Hill Construction*, em 2013, na qual é apresentada uma lista de benefícios relativos à implementação do BIM, referidos pelas empresas envolvidas no estudo como fazendo parte de um conjunto dos três (3) mais importantes para a sua organização.

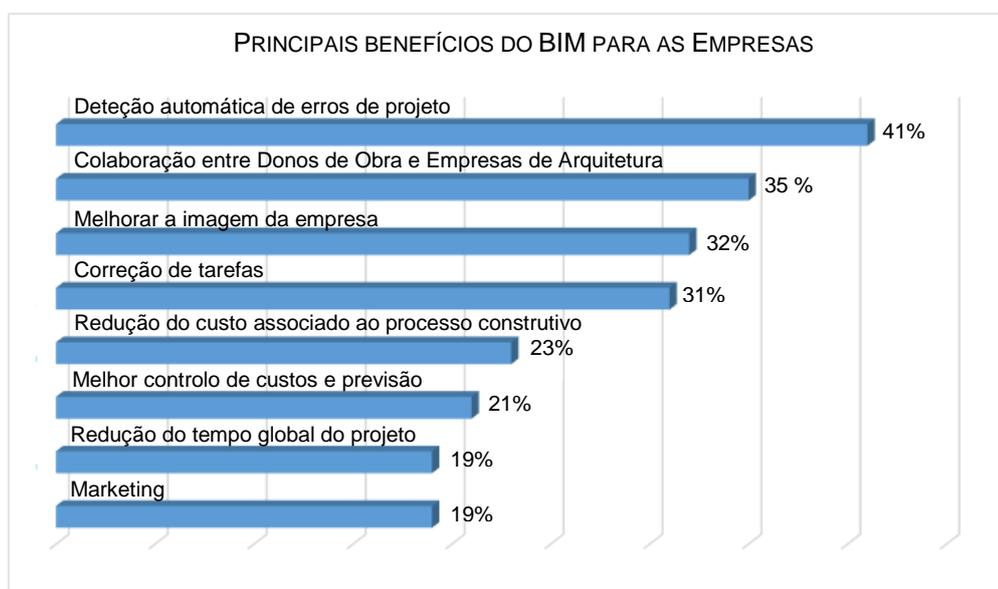


Figura 2.11 – Principais benefícios do BIM para as empresas. Adaptado de [36].

Pela sondagem anterior, constata-se que do conjunto dos benefícios apresentados, a generalidade tem impacto direto no retorno do investimento. Contudo, segundo uma outra sondagem, também da *McGraw Hill Construction*, o retorno que as empresas podem colher do BIM depende diretamente do nível de compromisso destas para com a mudança [36].

Apesar de existir sempre uma margem de risco associada ao retorno do investimento, a situação tende a melhorar caso as empresas apostem na mudança. O grau de compromisso para com a mudança pode influenciar as possibilidades de obtenção de um retorno muito positivo em 39% e de um retorno negativo em 25%, como se pode observar pela Figura 2.12 **Error! Reference source not found.** Acrescenta-se, ainda, que com a melhoria das competências dos profissionais e com a experiência da própria empresa no BIM, o retorno tende a ser mais elevado [36].

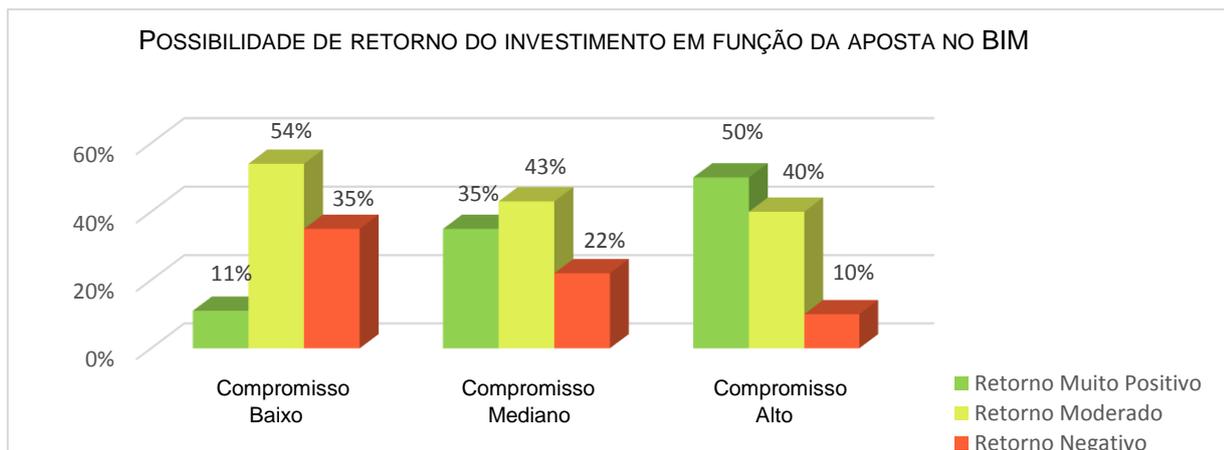


Figura 2.12 – Possibilidade de Retorno do Investimento em função da aposta no BIM. Adaptado de [36].

A adoção do BIM está a expandir um pouco por todo o mundo, sendo a procura de maior qualidade pelo menor custo possível o principal fator para este fenómeno. Na Figura 2.13 é exposto o ponto de situação de alguns países em relação ao estado de implementação do BIM.



Figura 2.13 - BIM no mundo (alguns países). Adaptado de [37].

Os Estados Unidos da América são um dos exemplos mais evidentes da mudança para o BIM, com um aumento de 28% para 71% entre 2007 e 2012. No entanto, o tamanho da sua indústria tem sido um obstáculo à aceitação global da mudança. Em contraste, a Finlândia com uma indústria construtiva de tamanho inferior é, atualmente, o país mais avançado relativamente à implementação do BIM, utilizando-o em projetos de pequena e grande escala. Existe, no entanto, uma tendência do BIM ser utilizado preferencialmente em projetos de grande escala. Em diversos países já foi deliberada a obrigatoriedade de implementação do BIM nas obras públicas, nomeadamente Finlândia, Noruega (pela conceção de diretivas e orientações), Singapura (com alterações nas leis), Dinamarca, China (utilizando o “*Industry Foundation Classes*”), Holanda (pela imposição de limites máximos ao custo da obra para

projetos independentes ao BIM ²) e, para o ano de 2016, o Reino Unido (com o objetivo de reduzir as emissões de carbono). Acrescenta-se, que a sua implementação em países caracterizados por uma indústria fragmentada é mais morosa, como é o exemplo da Alemanha e Portugal. Refere-se, ainda, que apenas foram referenciados alguns países que estão a apostar no BIM, não querendo menosprezar outros como o Japão, França, Canadá, Brasil, Coreia do Sul, Azerbaijão, entre outros [36], [37], [38]. Acrescenta-se que o papel desempenhado pelo governo de cada país é fundamental para o êxito da implementação do BIM [39].

Em Portugal, caso as entidades governamentais optassem pela obrigatoriedade de utilização de ferramentas BIM nos concursos públicos, as necessidades do setor poderiam ser assim colmatadas, dando, por isso, resposta à situação económica atual [38]. Além disso, as empresas Portuguesas apresentam níveis de competitividade inferiores quando comparadas com as empresas europeias. Esta diferença reflete-se no não cumprimento de prazos, na segurança ineficiente, na falta de qualidade e nos orçamentos ultrapassados [40]. Segundo o inquérito concretizado no âmbito do projeto de investigação apresentado em [40] e realizado a 109 donos de obra públicos, os inquiridos escolheram como medida mais significativa para a melhoria da qualidade dos processos construtivos, um aperfeiçoamento dos métodos comunicativos, pelo que o BIM poderá ser uma mais-valia nesse sentido.

É evidente que o BIM está associado a uma reestruturação global dos processos construtivos e, por isso, a sua implementação tem que ser faseada, de acordo com critérios que potenciem os resultados, isto é, que permitam alcançar a maturidade pretendida. Apenas assim é possível alcançar o objetivo a longo prazo do BIM, o *Integrated Project Delivery* (IPD) [41]. Este conceito constitui numa abordagem que pertence “integrar pessoas, sistemas, estruturas e práticas através de um processo colaborativo, explorando as aptidões/ capacidades de todos os intervenientes, de modo a otimizar os resultados do projeto, aumentar o valor para o proprietário, reduzir desperdícios e máximizá-la a eficácia de todas as fases de projeto” [42].

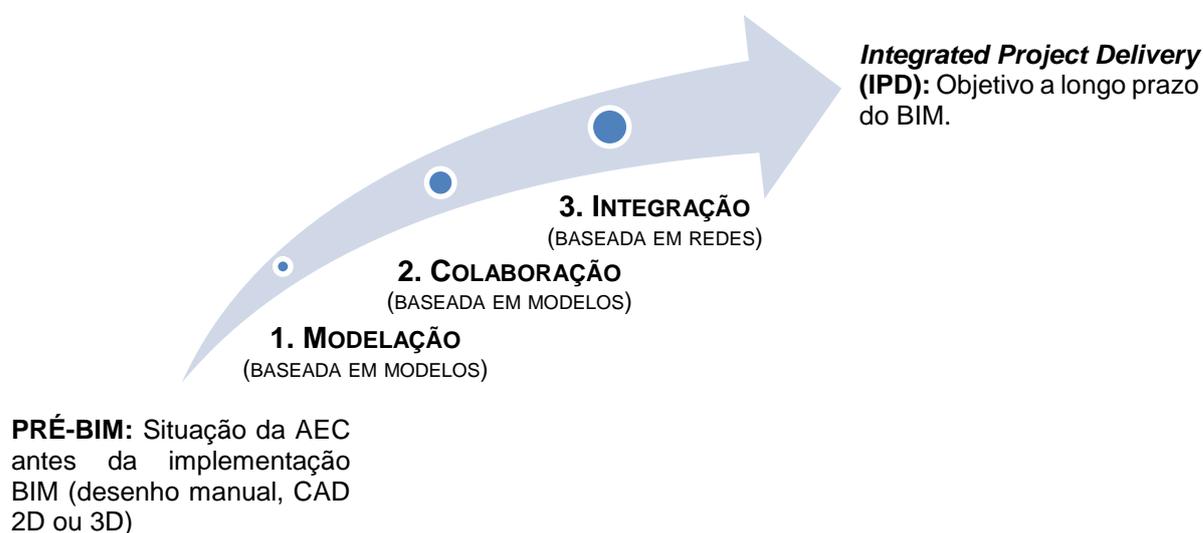


Figura 2.14 - Faseamento para a implementação do BIM. Adaptado de [38]

Na primeira fase, “Modelação”, as ferramentas BIM são utilizadas com o intuito de produzir visualizações tridimensionais (3D) e documentação geral. Posteriormente, na etapa “Colaboração”,

² Assim como a Holanda, também a Finlândia apresenta esta regra em vigor.

como o nome indica, os processos realizam-se segundo um ambiente colaborativo, com partilha de informações. É nesta fase que se praticam as tarefas referentes ao planeamento (4D) e à orçamentação (5D). Por fim, segue-se o processo baseado em redes, “Integração”, onde se constituem todas as dimensões adicionais do projeto através da integração de modelos, ao longo de todas as fases de uma obra, com o intuito de conceber um modelo único, universal e partilhado [38].

Um guia para a implementação do BIM, baseado em experiências práticas, foi disponibilizado pela *Associated General Contractors of America (AGC)*, em 2007, o qual se intitula “*AGC Contractors’ Guide to BIM*”, estando disponível em: [43]. Entretanto, este documento foi atualizado no ano de 2010.

Acrescenta-se que as empresas de grande dimensão têm que agir como mentoras das de pequena dimensão e encorajá-las a aceitar a mudança, adotando novos métodos de partilha da informação.

Apresentam-se na Figura 2.15 algumas das obras mais mediáticas do presente, nas quais o BIM foi escolhido como ferramenta de trabalho.



Figura 2.15 - Projetos BIM: a) Flame Towers, Azerbaijão [44]; b) Terminal Oeste de Kowloon, Hong Kong [45]; c) Estação subterranea London Victoria, Inglaterra [46]; d) Canal de Panamá, Panamá [47].

2.4. REGRAS DE MEDIÇÃO

Os atos de medição correspondem à contabilização de quantidades de tarefas ou artigos de orçamentação, os quais devem ser regidos por um conjunto de regras, sendo que estes podem ser efetuados sob o projeto ou na obra [48]. Com efeito, uma estimativa orçamental pode ser dividida em duas parcelas, uma referente às quantidades de trabalhos que dependem do profissional responsável por cada tarefa e uma outra relativa à produtividade dos trabalhadores e equipamentos empregues [49]. O documento onde todas estas informações estão reunidas, denomina-se Mapa de Trabalhos e Quantidades.

Ao introduzir a temática das regras de medição, há indispensavelmente que fazer referência aos documentos normativos aplicáveis ao território nacional.

O Decreto-Lei n.º 18/2008 de 29 de Janeiro [50] aprova o Código dos Contratos Públicos (CPP), documento que “estabelece a disciplina aplicável à contratação pública e o regime substantivo dos contratos públicos que revistam a natureza de contrato administrativo”. Na subsecção I da seção VIII do referido DL são expostos todos os assuntos alvo de análise no âmbito de atos de medição. De acordo com o ponto 3 do Artigo 388.º, presente nesta subsecção, “Os métodos e critérios a adotar para realização das medições devem ser definidos no contrato”.

A Portaria n.º 959/2009 de 21 de Agosto [51] corresponde ao formulário do caderno de encargos proposto no Decreto-Lei n.º 18/2008. Este formulário é composto pelas cláusulas gerais, que poderão servir de base para a formulação de contratos de empreitadas referentes a obras públicas. No ponto 3 da Cláusula 26.ª da Seção III, referente às medições de obra, é indicado que em alternativa ao Artigo 388.º, referido anteriormente: “ (...) Os métodos e os critérios a adotar para a realização das medições respeitam a seguinte ordem de prioridades [indicar outros critérios, se for o caso]:

- a) As normas oficiais de medição que porventura se encontrarem em vigor;
- b) As normas definidas no projeto de execução;
- c) As normas definidas pelo Laboratório Nacional de Engenharia Civil;
- d) Os critérios geralmente utilizados ou, na falta deles, os que forem acordados entre o dono da obra e o empreiteiro.”

As alternativas representadas pelas letras “a” e “c” não existem na prática, sendo usual admitir para as normas representadas pela letra “c”, os critérios definidos na publicação “Medições na Construção de Edifícios”. Este documento resulta de um estudo primário levado a cabo pelo LNEC, cuja primeira publicação data de 1987. Este conjunto de regras foi concebido com o intuito de simplificar e padronizar os métodos de análise praticados pelos intervenientes e examinado por um conjunto de indivíduos constituído por especialistas e entidades públicas e privadas. A última atualização deste documento é apresentada no livro “Curso sobre Regras De Medição Na Construção”. [52]

No âmbito das medições em Portugal é interessante fazer referência ao projeto ProNIC (Protocolo para a Normalização da Informação Técnica na Construção). Esta iniciativa tem como objetivo a criação de bases de dados de conhecimento constituídas por especificações técnicas, regras de medição, fichas de custos, fichas de execução de trabalhos, fichas de materiais e regras de segurança. Por sua vez, estas bases de dados são articuladas a uma plataforma eletrónica da construção para a gestão da informação, garantindo assim a presença de toda a informação necessária. A sua aplicabilidade abrange edificações novas ou de reabilitação e infraestruturas rodoviárias. O seu desenvolvimento é conseguido através de um consórcio composto pelo Laboratório Nacional de Engenharia Civil (LNEC), pelo Instituto da Construção (IC-FEUP) e pelo Instituto de Engenharia de Sistemas e Computadores do Porto (INESC) [53]–[55].

2.5. BIM NO CONTEXTO DE MEDIÇÕES

Primeiramente, para relacionar o BIM com medições é indispensável ter em conta que as medições regem-se por um conjunto de regras bem estruturadas, as quais devem ser respeitadas ao pormenor pelo modelador de modo a conceberem-se modelos concisos e corretos. A utilização do BIM para obtenção automática de quantidades proporciona não só oportunidades, mas também desafios para a profissão de gestor de obra [29].

A utilização apropriada do BIM para elaboração de estimativas conduz a estimativas mais exatas (*precise*) e certas (*accurate*). Estas designações apesar de semelhantes são distintas. Enquanto a primeira, refere-se ao grau de refinamento utilizado para a elaboração da estimativa, a segunda, está relacionada com a aproximação da estimativa ao custo real [49]. Um dos maiores obstáculos a esta quinta (5ª) dimensão do BIM consiste na qualidade associada aos modelos virtuais, uma vez que estes são compostos por uma enorme quantidade de informação interconectada de forma complexa e introduzida manualmente. Neste sentido, há que minimizar a introdução incorreta ou inapropriada de informação [29]. O BIM, como ferramenta neste processo, representa um papel crucial na redução de erros aleatórios, ou seja, erros que derivam das capacidades interpretativas do medidor. No entanto, há também que considerar os erros sistemáticos, i.e. provenientes de informações fornecidas e incorretas para introdução no modelo virtual [49]. Portanto os erros das estimativas que, nos processos tradicionais, podem derivar da interpretação pessoal dos medidores, tornam-se responsabilidade do modelador.

A quantificação automática de quantidades através de modelos BIM, processo este designado em inglês por *Quantity takeoff*, é atualmente uma das aplicações com maior popularidade e utilidade do BIM. Uma grande parte das ferramentas BIM disponíveis no mercado mundial já possibilita a contabilização automática de quantidades através das propriedades geométricas dos diferentes elementos contidos no modelo. Com estas propriedades são executados cálculos internos para determinação das áreas superficiais e dos volumes associados [56]. Contudo, estas ferramentas não possibilitam a manipulação da informação para estimar os custos associados, sendo para tal necessário recorrer a *softwares* externos os quais operam com bases de dados de custos unitários [56], [57]. As trocas de informações entre estes *softwares* pode ser conseguida se o formato em que estes operarem for o mesmo ou, em alternativa, utilizando o IFC como formato padrão [56]. Note que, recorrendo à última opção apresentada, poderão ocorrer erros provenientes de eventuais perdas de informação, uma vez a conversão de modelos entre ferramentas BIM e o IFC ainda não são perfeitas [58]. Na Figura 2.16 é apresentado o processo para a elaboração do Mapa de Trabalhos e Quantidades.

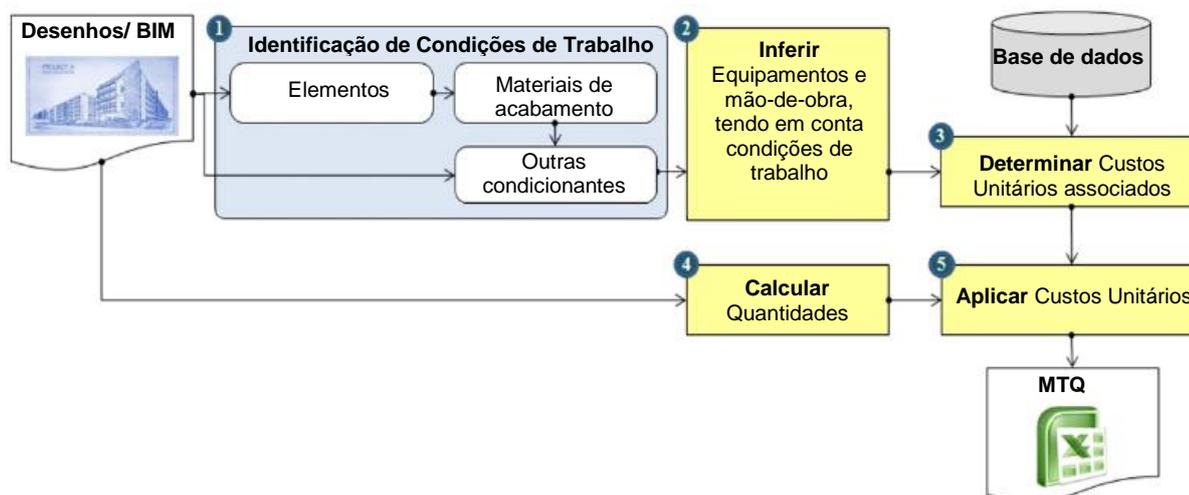


Figura 2.16 - Processos para estimação de custos. Adaptado de [57].

O *Autodesk Revit*, o *Graphisoft ArchiCAD*, o *Vico Office Takeoff Manager*, o *Solibri Model Checker* e *CostX* são alguns exemplos de ferramentas que apresentam capacidades de extração automática de quantidades. Acrescenta-se, ainda, que a *Autodesk* desenvolveu dois programas adicionais que apresentam esta função, nomeadamente o *Autodesk Naviswork* e um outro, concebido apenas para este fim, intitulado *Autodesk Quantity Takeoff*.

Se o material empregue numa determinada obra for betão armado, então, as parcelas referentes aos elementos contabilizados nos mapas de trabalhos e quantidades dividem-se em volumes, cofragens e armaduras. A obtenção de volumes de materiais é alcançada diretamente através dos programas apresentados. No entanto, tal não sucede para as cofragens, não existindo sequer nenhuma ferramenta que permita modelá-las manualmente. Relativamente às armaduras, estas são incorporadas no modelo virtual recorrendo a um programa com uma vertente estrutural [56].

Enquanto para o contratante, a obtenção de estimativas cada vez mais certas é fundamental, para os arquitetos e engenheiros envolvidos, o custo e esforço que tal acarreta não é, muitas vezes, compensado [29].

Em suma, esta característica do BIM proporciona um aumento da precisão das estimativas nas fases iniciais do projeto, quando comparado com os processos tradicionais, uma redução do tempo e custo para concretizar essas estimativas e uma maior facilidade de simulação/ teste de diferentes alternativas [49].

“Rather than a threat, BIM and automated quantities technologies provide the profession with enormous opportunities to raise the value of their services to a much higher and sophisticated level. It is not simply about automatic quantities generation.”

Dr. Peter Smith

2.6. IFC

2.6.1. CONCEITO

Para potenciar a comunicação entre diferentes ferramentas BIM que não partilham formatos nativos, procura-se a utilização de uma linguagem comum, que proporcione a interoperabilidade necessária entre estas ferramentas. Esta “linguagem-mãe”, que se apresenta em evolução constante e em formato aberto, é o *Industry Foundation Classes (IFC)*. A *International Alliance for Interoperability (IAI)*, presentemente designada por *buildingSMART Alliance*, é uma instituição internacional fundada em 1994 e formada por indivíduos do setor da construção, sem qualquer fim lucrativo, que procura desenvolver esta linguagem [59]. Com o desenvolvimento do IFC pretende-se melhorar diversas valências no setor da Construção Civil, nomeadamente os processos de comunicação, a produtividade, o cumprimento de prazos, a redução de custos e o aumento da qualidade durante todas as fases do ciclo de vida de uma edificação [60].

O IFC insere-se no conceito de *OpenBIM*, o qual corresponde à criação de uma “nuvem” na qual são adicionados todos os modelos BIM referentes às mais diversas dimensões, em formato IFC. Toda e qualquer alteração nas informações enviadas apenas pode ser executada pela ferramenta que as concebeu. Pretende-se, portanto, combinar modelos específicos de modo a coexistirem durante todo o projeto [61].

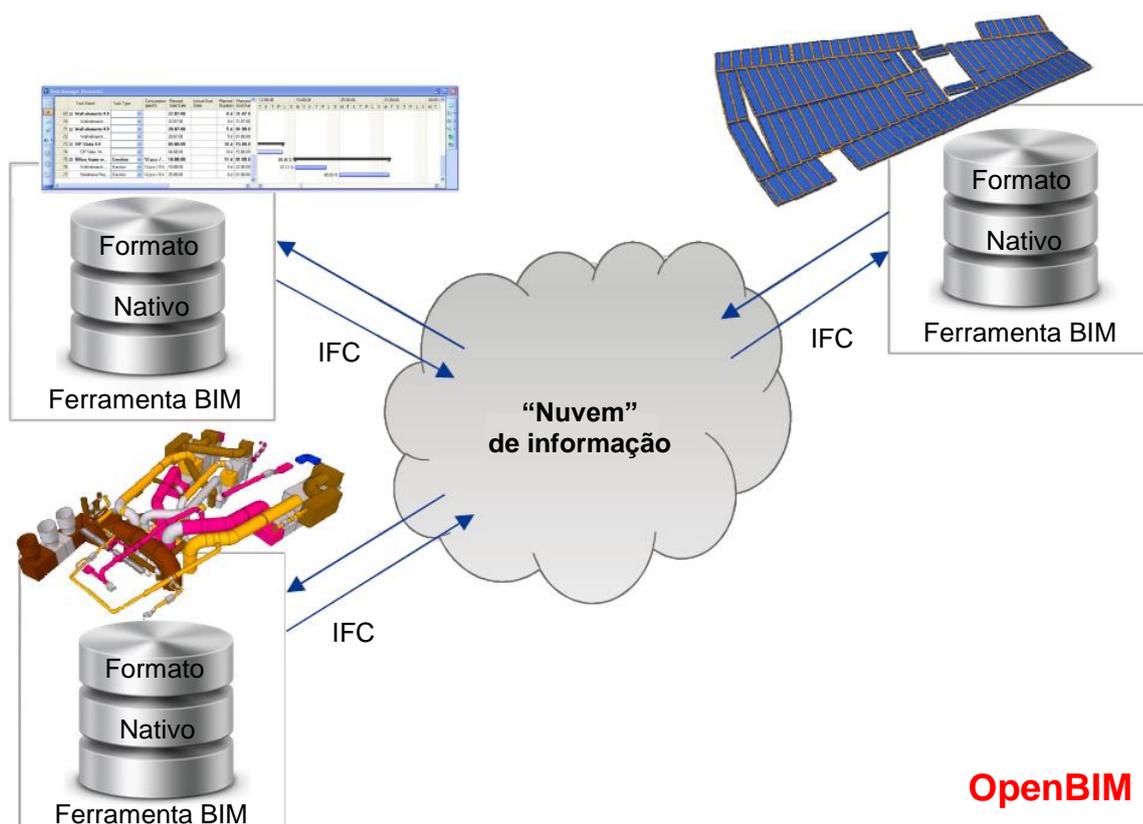


Figura 2.17 - IFC no conceito de OpenBIM. Adaptado de [61]

O formato IFC pretende constituir formas neutras de transposição de toda a informação contida num modelo BIM relativa a todas as dimensões por este suportadas, incluindo interdependências entre informações. Por outras palavras, o IFC permite a constituição de meios para o desenvolvimento de um modelo BIM, já existente, a partir de um *software* diferente daquele onde foi concebido, o qual, pode ser novamente exportado para IFC.

Atualmente existem três tipos de formatos IFC, nomeadamente IFC-SPF, IFC-XML e IFC-ZIP, sendo sobre o primeiro que esta dissertação incide (IFC-SPF). Este formato pode ser identificado pela extensão "*.ifc" [59]. Os conceitos base nos quais o IFC-SPF se baseia podem ser consultados no *site* da *BuildingSMART* [59] e na Norma ISO 10303-21 (*Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange*), a qual pretende instituir procedimentos padrão para a representação, interpretação e troca de informações em linguagem EXPRESS [62]. Esta linguagem é uma linguagem orientada a objetos, onde os objetos podem ser relacionados entre si.

A Norma ISO 10303 é também conhecida por "STEP", ou seja, "*Standard for the Exchange of Product model data*". Esta é subdividida em partes distintas, sendo cada uma destas caracterizada por um tipo de estrutura de dados e direcionada a um determinado setor da indústria. A cada uma destas partes atribui-se a designação Protocolos de Aplicação. Como referido anteriormente, é sob a estrutura do Protocolo de Aplicação número 21 que o IFC-SPF se encontra enquadrado. A sua estrutura é do tipo "STEP-File" (consiste num tipo de troca de informação "STEP") caracterizada por uma estrutura do tipo ASCII (esquema de codificação de caracteres baseado no alfabeto Inglês), no qual é apresentado normalmente uma instância por linha [63].

2.6.2. BREVE HISTÓRIA

O IFC começou a ser desenvolvido no ano de 1994 aquando da constituição da *International Alliance for Interoperability* (IAI) e em 1997 surge a sua primeira versão, nomeadamente “IFC 1.0”. Logo após a publicação desta versão, 26 empresas mundiais (entre as quais a *Autodesk*, *Bentley*, *IEZ* e *Nemetschek*) mostraram interesse na implementação desta funcionalidade nas suas próprias ferramentas [64]. Seguiram-se as versões apresentadas na cronologia da figura seguinte até à mais recente atualização, o IFC 4, onde cada vez mais são disponibilizadas novas funcionalidades como, a título de exemplo, quantidades padronizadas para medições, análises estruturais e detalhe, análise energética, entre outras [65]. Contudo, no presente, esta última versão ainda tem um âmbito de aplicação muito limitado, uma vez que são poucos os *softwares* que o suportam, estando a maioria das ferramentas BIM a operar com o IFC 2x3 lançado no ano de 2007.

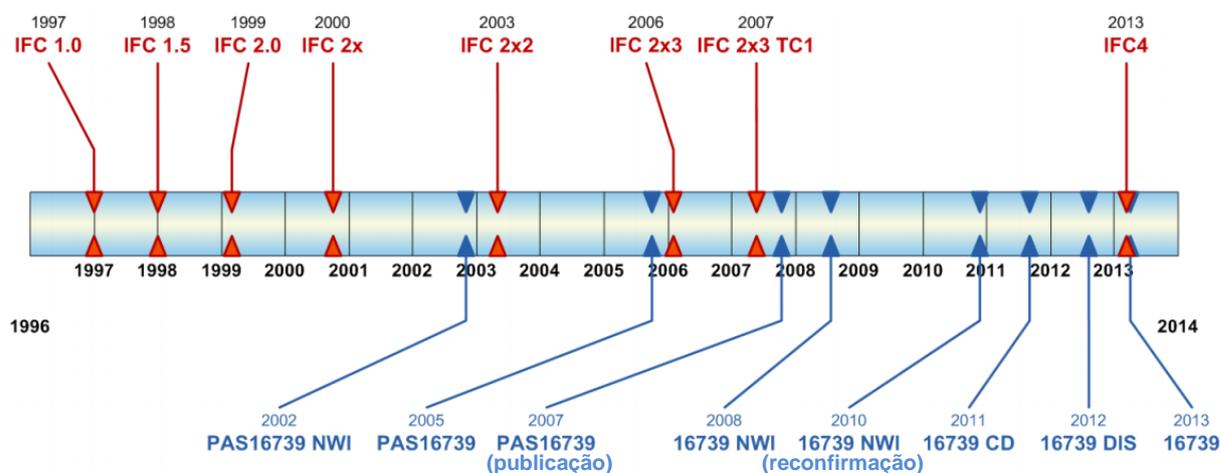


Figura 2.18 - História do IFC. Adaptado de [65].

2.6.3. ESTRUTURA/ ARQUITETURA DO IFC

O IFC apresenta uma estrutura com inter-relacionamento de informação. O modo como esta informação é disposta é diretamente dependente da versão IFC para a qual o modelo BIM é convertido. Com o IFC, todas as informações referentes às especialidades do sector da construção são acopladas numa única base de dados do tipo texto. O desafio consiste em reunir todas estas informações sem criar erros ou conflitos [66].

A arquitetura concebida para o modelo IFC baseia-se em estruturas modelares que permitem o desenvolvimento de diversas componentes [67]. A constituição de uma arquitetura para o modelo IFC tem como principais objetivos proporcionar uma estrutura base que permita o funcionamento do modelo, promover um ambiente colaborativo com trocas de informações entre os mais diversos intervenientes no processo construtivo, viabilizar a reutilização de modelos, simplificar a manutenção da especificação, conduzir à compatibilidade entre diferentes ferramentas e, finalmente, permitir a utilização de componentes por parte de distribuidores de *software* [68].

Os princípios que estão na base desta arquitetura são ilustrados na Figura 2.19. Salienta-se que uma classe pode referenciar uma outra situada ou na mesma camada ou numa camada inferior, no entanto, não pode referenciar uma camada superior.

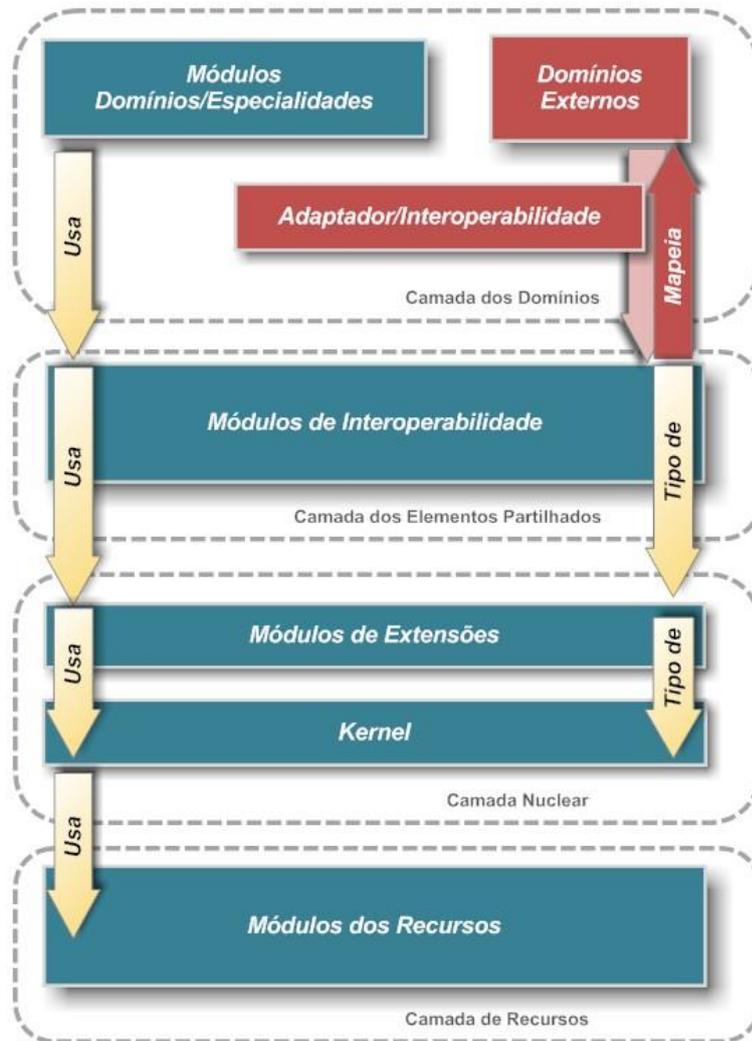


Figura 2.19 - Estrutura do IFC [69]. Adaptado de [70].

A sua estrutura apresenta uma hierarquia bem definida e composta por quatro (4) camadas distintas, nomeadamente [66], [68], [70]:

- **Camada de recursos** – abrange os esquemas de dados que permitem o suporte das estruturas de dados do modelo. Constitui a camada mais baixa na arquitetura da especificação IFC e é nesta que são definidas entidades genéricas que possibilitam formular um conjunto de dados para a construção de modelos;
- **Camada Nuclear** – constitui a camada mais geral num esquema IFC, englobando funções que sustentam o modelo IFC e a sua estrutura básica, assim como relações fundamentais para a especificação de modelos;
- **Camada dos elementos partilhados** – responsável pelas questões inerentes à interoperabilidade de sistemas. Compreende as entidades referidas em vários módulos da camada de domínios;
- **Camada dos domínios** – esta camada identifica o âmbito de uma especificação IFC. A indústria AEC é dividida em grupos de especialidades, com total independência das entidades pertencentes à camada dos domínios, não podendo ser referenciada por outras camadas.

2.7. WEB SERVICE

2.7.1. CONCEITO

A comunicação é um fator chave para o sucesso. Os *Web services* surgem como um meio de disponibilização de serviços que possam ser acedidos e partilhados por diferentes pessoas, aplicações e mesmo outros *Web services*, com um custo reduzido. Portanto, os *Web services* são ferramentas, acedidas via internet, que utilizam estruturas baseadas em *standards* abertas como o XML (*eXtensible Markup Language*) e o HTTP (*Hypertext Transfer Protocol*) e são independentes de *hardware*, sistemas operativos e linguagens programáticas [71]. O protocolo HTTP constitui um suporte para a comunicação da informação através da *Web*, permitindo uma relação bidirecional entre emissores e recetores [72].

Os *Web services* podem-se dividir em dois grandes grupos, tendo em conta a forma como a sua implementação é concretizada, nomeadamente os “*Big*” e os “*RESTful*” *Web services* [73]. Uma vez que o *Web service* projetado na presente dissertação pertence ao segundo grupo exposto, apresentam-se, de seguida, contornos gerais relativos à sua estrutura.

2.7.2. WEB SERVICE DO TIPO “RESTFUL”

O termo “*RESTful*” deriva da tecnologia REST (*Representational State Transfer*) e representa a sua implementação prática. Esta tecnologia não representa um protocolo nem uma *standard*, mas apenas um estilo arquitetónico que utiliza *standards* bem instituídas da W3C (*World Wide Web Consortium*), nomeadamente HTTP, XML, URI e MIME [74]:

- HTTP (*Hypertext Transfer Protocol*). Constitui um “protocolo para sistemas de informação de hipermídia distribuídos e colaborativos” [75], ou seja, um protocolo direcionado para a transferência de dados entre computadores;
- XML (*eXtensible Markup Language*). É empregue para a descrição da informação através de estruturas hierárquicas, onde podem ser adicionadas informações respeitantes aos dados de passagem. Esta característica permite ao utilizador compreender o que cada informação representa. Caracteriza-se pela sua simplicidade e interoperabilidade [71];
- URI (*Uniform resource identifier*), como o nome indica, constituem um conjunto de caracteres identificativos de recursos, como por exemplo URL’s (endereços eletrónicos) [76];
- MIME (*Multipurpose Internet Mail Extensions*). Esta extensão permite a partilha de informação em diferentes formatos (áudio, texto, imagem, vídeo, entre outros), constituir mensagens com multiobjectos e de tamanho ilimitado e a utilização de caracteres além do tipo ASCII [77].

A utilização de tecnologias do tipo REST caracteriza-se pela sua maior simplicidade de comunicação em relação a outras, uma vez que apresenta uma melhor interação com protocolos da *Web* (HTTP) [74], [75]. Um *Web service* contém uma API (*Application Programming Interface*), a qual representa um conjunto de regras relativas ao funcionamento do programa, e a respetiva documentação (da API) para a utilização do *Web service*.

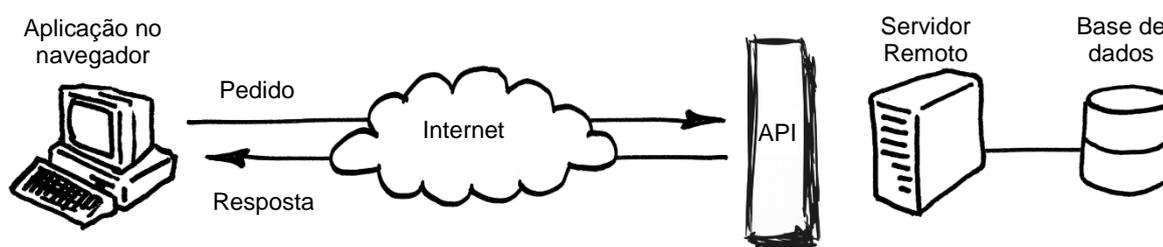


Figura 2.20 – Processo de funcionamento de um *Web service*. Adaptado de [79].

Apresenta-se de seguida um exemplo tipo de um *Web service* do Google, respeitante à pesquisa de informações:

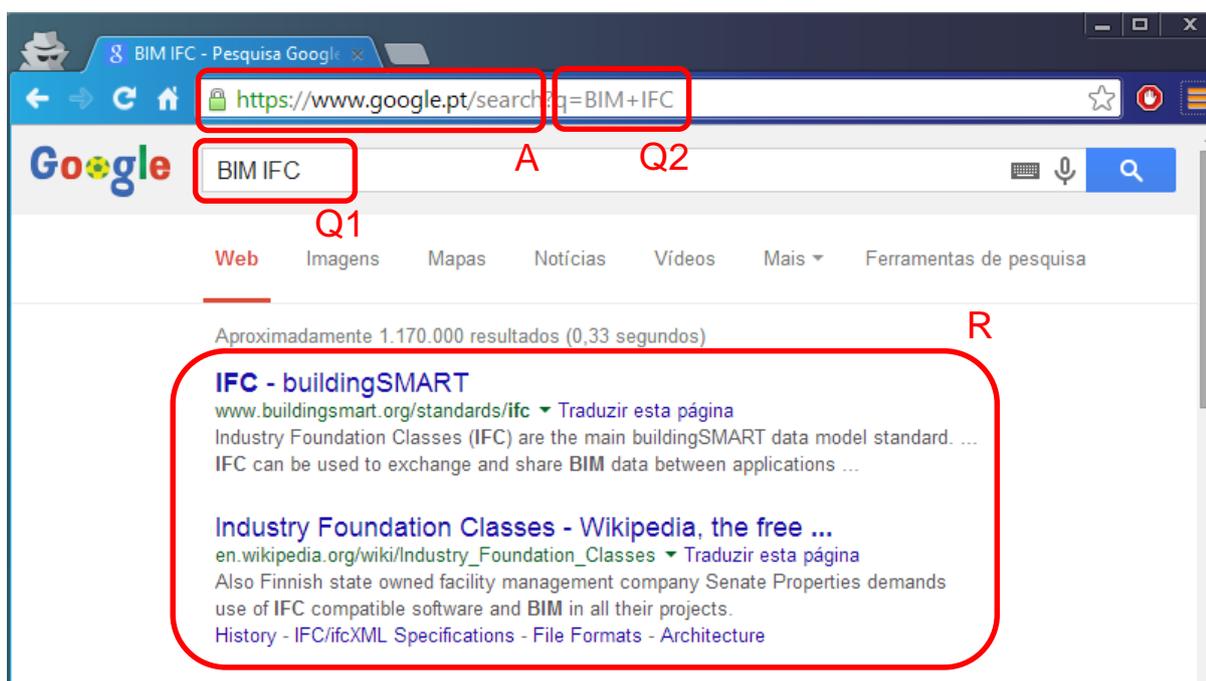


Figura 2.21 - Exemplo de uma pesquisa no Google e resposta HTML. Adaptado de [80].

Na figura anterior, destacam-se quatro (4) campos distintos, nomeadamente a localização do *Web service* do Google (A), a pesquisa realizada no caixa de pesquisa (Q1), a pesquisa realizada via URL (Q2) e a resposta ao pedido (R). A resposta do serviço é apresentada sob a forma de HTML (*Hyper Text Markup Language*) após o processamento dos algoritmos envolvidos no processo [80]. Neste exemplo, a utilização do *Web service* de pesquisa do Google é simples, no entanto, tal nem sempre sucede e, por isso, é correntemente disponibilizada documentação referente à API para a utilização do *Web service*.

2.7.3. VANTAGENS

Os benefícios latentes à constituição e implementação *Web services* são evidentes e são apresentados de seguida [71], [81], [82]:

- a. Baseados em normas abertas;
- b. Utilização simples comparativamente a outras soluções;
- c. Acessíveis via Internet;
- d. São polivalentes uma vez que podem ser acedidos por humanos, aplicações e *Web services*;
- e. Permitem a unificação da informação, possibilitando a partilha de informações entre diferentes aplicativos;
- f. Possibilitam a partilha de serviços existentes e disponibilizados por terceiros, o que suprime a necessidade de constituição de ferramentas que proporcionem serviços equivalentes;
- g. A utilização de diferentes linguagens programáticas entre diferentes aplicativos não é um obstáculo;
- h. O investimento tecnológico é reduzido uma vez que utiliza a Internet como ferramenta neste processo;
- i. O custo associado a estas ferramentas é globalmente reduzido devido a todas as vantagens apresentadas.

3

EXTRAÇÃO DE QUANTIDADES SUPPORTADA POR BIM

3.1. DESCRIÇÃO GERAL

Pretende-se neste Capítulo efetuar uma análise detalhada ao modelo IFC, de modo a inferir se através das relações estabelecidas por esta linguagem é possível associar materiais a elementos construtivos e a estes, as suas propriedades geométricas, com o intuito de determinar quantidades de betão armado. Para tal, é modelada uma estrutura tipo através de diversas ferramentas computacionais, extraindo o seu modelo IFC e estudando os comportamentos das parcelas relevantes para o presente estudo.

3.2. REGRAS DE MEDIÇÃO

As regras de medição consideradas nesta dissertação para a modelação de estruturas, como já referido, são definidas a partir do ponto de vista da última publicação/ atualização “Curso sobre Regras De Medição Na Construção”.

O primeiro documento preliminar referente às regras de medição realizado em Portugal pelo LNEC, na data de 1970, com o nome de “Regras de Medições” foca um aspeto interessante que é transcrito de seguida: “Os métodos adotados são, em geral, baseados na interpretação pessoal dos medidores, muitas vezes mal preparados, o que origina, com frequência, ambiguidades e deficiências nas diferentes etapas de realização dos empreendimentos de edifícios” [83]. Apesar de em pleno século XXI estes obstáculos estarem praticamente ultrapassados, fruto da constituição de bases sólidas para a elaboração de medições (ainda que continue a não existir uma norma portuguesa), a citação transcrita faz todo o sentido quando enquadrada com a mudança de paradigma de processos manuais para o BIM.

Quer-se com isto dizer que, atualmente, um indivíduo habilitado para quantificar tarefas analisará as peças desenhadas, as condições técnicas do projeto e os trabalhos realizados em obra e adotará as regras de medição previamente estipuladas. Numa perspetiva de obtenção automática de quantidades via IFC, apesar de ser possível a constituição de um modelo 3D com as informações que constituem estes ficheiros, é inviável a imposição de regras de medição padrão ao modelo IFC através de uma ferramenta de cálculo automático. Tal deve-se à complexidade envolvida na readaptação da informação base, considerando interdependências entre elementos. Com efeito, as medições apenas terão em conta o modelo IFC gerado sem readaptação da informação base. Acrescenta-se que qualquer informação externa ao modelo IFC não poderá ser considerada neste processo.

Por conseguinte, todas as regras que deverão ser adotadas para medições deverão ser respeitadas na fase de modelação e não numa posterior. Apenas desta forma, dando o uso apropriado à tecnologia

atualmente disponível, é possível evitar a problemática da quantificação de trabalhos com critérios díspares.

Na modelação computacional, se não existissem regras base a seguir pelo modelador, este poderia adotar um conjunto de diferentes abordagens ao projeto. Consequentemente pôr-se-ia em causa a coerência de todo o trabalho desenvolvido. As dissemelhanças físicas entre diferentes abordagens poderiam nem ser evidentes à primeira vista, no entanto, passariam a sê-lo aquando da utilização dos dados referentes a cada elemento estrutural.

Por exemplo, sem um conjunto de diretrizes que permita definir quais os elementos que prevalecem em interseções de vários elementos estruturais, poderão obter-se medições não coerentes. Na Figura 3.1, apresenta-se, a título de exemplo, diferentes abordagens à intersecção de uma viga com um pilar.

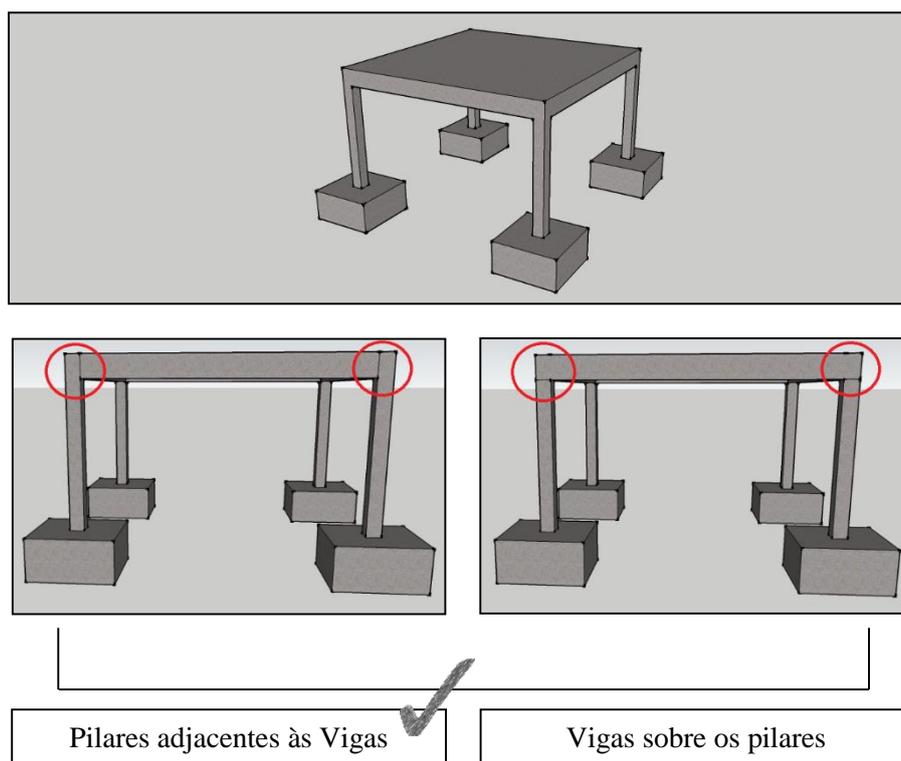


Figura 3.1 - Diferentes abordagens para a intersecção: Pilar/ Viga.

Na figura anterior, segundo a publicação referente às regras de medições a adotar, a abordagem correta para a quantificação de quantidades é a modelação dos pilares adjacentes às vigas. Já para a laje, o volume a adotar é aquele obtido considerando o comprimento e largura entre as faces das vigas e pilares.

As medições de betão armado englobam três componentes distintas, nomeadamente volumes, cofragens e armaduras. Contudo, este estudo refere-se apenas à obtenção de volumes de elementos estruturais comuns à generalidade dos edifícios em betão, estabelecendo metodologias para a sua obtenção.

O ponto de vista desta dissertação apenas abrange medições relativas aos elementos estruturais que compõem a obra, excluindo portanto todas as medições relativas ao estaleiro (i.e. pessoal, instalações provisórias e equipamentos), trabalhos preparatórios de execução da obra (como por exemplo desmatização e proteções para a segurança), demolições, movimentos de terras, pavimentos e drenagens exteriores. Pretende-se instituir uma abordagem inicial à temática das medições.

3.3. IFC (INDUSTRY FOUNDATION CLASSES)

3.3.1. LINGUAGEM E SIMBOLOGIA

Como já foi referido existem três tipos de formatos IFC, nomeadamente IFC-SPF, IFC-XML e IFC-ZIP, sendo o primeiro objeto de estudo na presente dissertação. Este pode ser identificado pela extensão “*.ifc”. Apresentam-se, de seguida, noções base relativas à estrutura tipo do IFC-SPF que permitam ao leitor compreender o trabalho desenvolvido. Primeiramente, ao abrir qualquer ficheiro IFC-SPF em formato de texto (“*.txt”) a primeira e última linha são sempre definidas por “ISO-10303-21;” e “END-ISO-10303-21;”, respetivamente. O conteúdo apresentado entre estas duas linhas divide-se em duas secções: “HEADER” e “DATA”.

Na secção “HEADER”, equivalente a um cabeçalho, introduzem-se informações gerais acerca do ficheiro sob o qual se está a operar (p.e. nome, autor, data de criação, descrição, tipo de esquema EXPRESS, entre outras), as quais podem ser divididas nos seguintes grupos: “file_description”, “file_name”, “file_schema”, “schema_population”, “file_population”, “section_language” e “section_context”. Refere-se a obrigatoriedade de existir uma instância de cada um dos três primeiros grupos referidos e segundo a ordem exposta.

Na secção “DATA” são integradas todas as informações do modelo gerado, as quais são dispostas segundo entidades/ classes. A disposição da informação respeita o esquema EXPRESS especificado na secção “HEADER”. Refere-se ainda que cada uma destas duas secções termina com uma linha contendo a terminologia “ENDSEC;” [62]. Na Figura 3.2 é exposta a estrutura tipo de um ficheiro IFC-SPF.

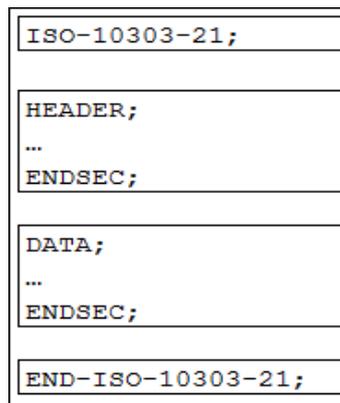


Figura 3.2 - Estrutura de um Ficheiro do tipo "STEP-FILE" [84].

Todas as entidades ou classes que compõem a secção “DATA” são constituídas pela sigla IFC, variando apenas a nomenclatura que se segue (p.e. “IFCORGANIZATION” e “IFCAPPLICATION”). Evidentemente, cada entidade apresenta uma função distinta.

Além disso, a cada entidade corresponde um indicador único, sendo este o primeiro número apresentado em cada linha e precedido por um cardinal (p.e. #1). Entre o indicador único e a respetiva entidade é utilizado um sinal igual (resultando “#1= IFCORGANIZATION”). Realça-se que a parcela numérica do indicador único é obrigatoriamente inteira positiva e pode atingir no máximo 9 dígitos.

Cada entidade é composta por um determinado número de atributos, onde se incorporam as informações respeitantes ao modelo. Estes podem variar de entidade para entidade segundo uma ordem específica, representam-se entre parêntesis e são separados por vírgulas. Após os atributos, segue-se um ponto e vírgula (p.e. “#1= IFCORGANIZATION (\$,'Autodesk Revit 2014 (ENU)',\$,,\$,);). Note que qualquer texto (string) incluído nos atributos é representado entre plicas (p.e. 'Autodesk Revit 2014 (ENU)'). Mais

ainda, sempre que um atributo não tiver associado a nenhum tipo de informação, este é representado pelo símbolo “\$”. O raciocínio aqui exposto pode ser facilmente perceptível através da Figura 3.3.

Refere-se que toda a informação referente a entidades e atributos, que depende também da versão IFC segundo a qual se opera, é disponibilizada pela *buildingSMART* através de uma biblioteca de informação.

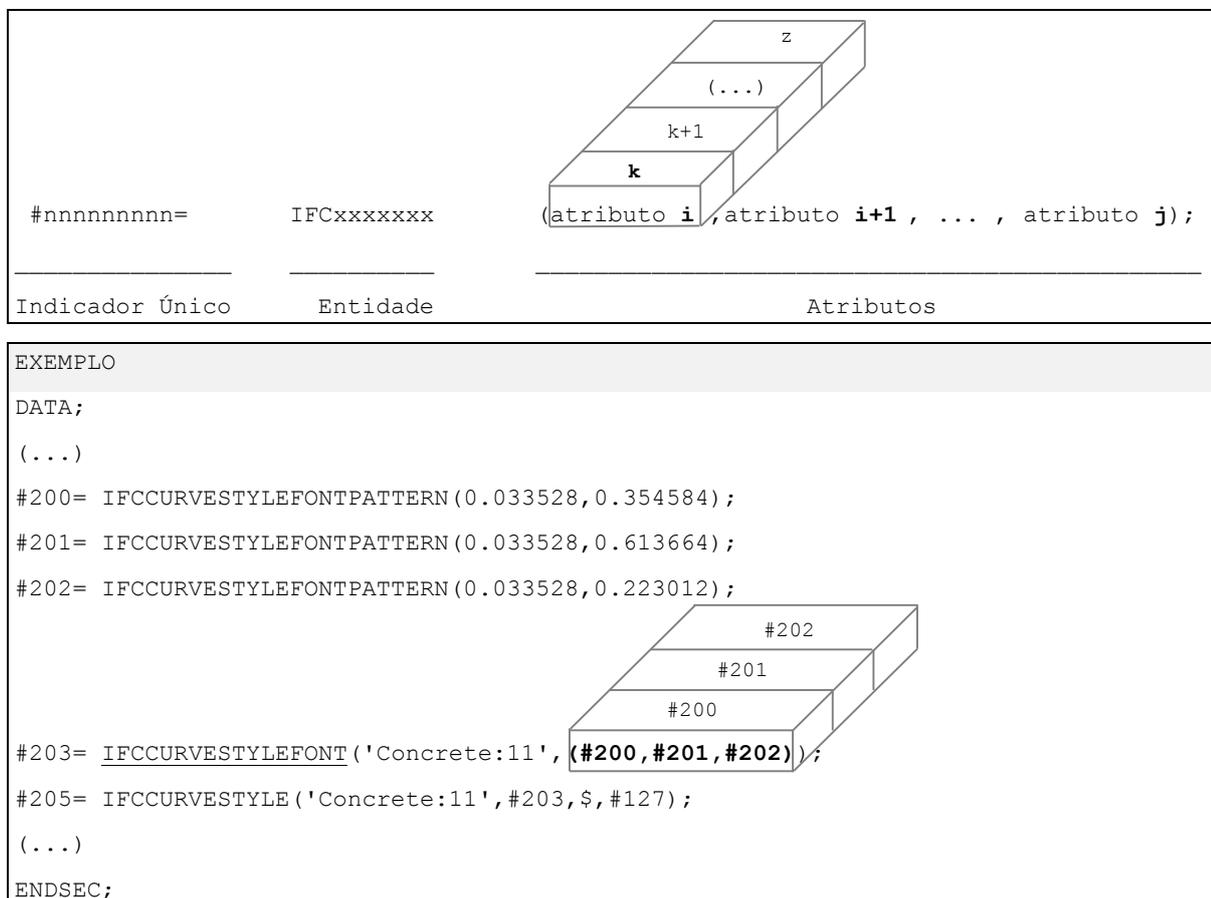


Figura 3.3 – Indicadores únicos, entidades e atributos. Adaptado de [84].

Resta referir que um atributo de uma qualquer entidade pode conter diferentes informações (“subatributos”), estando estas separadas também por vírgulas e entre parêntesis. Por exemplo, na figura anterior, a entidade “*IfcCurveStyleFont*” é composta por dois atributos, sendo o segundo composto por três informações distintas. Neste caso, essas informações são referências a outros indicadores únicos.

3.4. CASOS DE ESTUDO

3.4.1. ENQUADRAMENTO GERAL

Primeiramente, um código em IFC pode ser gerado por diferentes ferramentas computacionais. A sua conceção através destas pode conduzir a uma quantidade de informação excessiva comparativamente à sua construção manual. Salienta-se que a informação contida nos ficheiros IFC gerados através destes *softwares* é disposta de um modo desorganizado.

Além disso, cada ferramenta computacional é programada de uma forma distinta e, por isso, ao converter-se um mesmo modelo para formato *.ifc através de diferentes *softwares*, obtêm-se ficheiros IFC distintos podendo, inclusive, ocorrer variação da informação por omissão e predefinição automática de dados.

Há ainda que referir que no momento de exportação do modelo podem ser escolhidas diferentes versões IFC mediante disponibilidade do programa em utilização. Estes modelos exportados divergem de acordo com a respetiva finalidade (p.e. um modelo para análise estrutural, dito modelo analítico, será diferente de um modelo arquitetónico e de um modelo alusivo a instalações de edifícios).

As versões IFC em análise na presente dissertação são IFC 2x3 e IFC 4, sendo estas a penúltima e última versões disponibilizadas, respetivamente. A utilização do IFC 2x3 é justificada pela maioria das ferramentas BIM existentes no mercado ainda operar segundo as suas diretrizes. Aliás, até ao momento, apenas existe uma ferramenta computacional que possibilita a utilização do IFC 4, a qual se denomina *Constructivity*. Contudo, a sua versão gratuita é bastante limitada pois não possibilita algumas funcionalidades como, a título de exemplo, a exportação para IFC.

Neste sentido, e no âmbito desta dissertação, será relevante não só efetuar comparações entre modelos estruturais para as versões IFC enunciadas, analisando o modo como a informação é gerada, mas também investigar se o modelo para cálculo estrutural, modelo analítico, é constituído pelas mesmas propriedades geométricas que um modelo arquitetónico. Tais análises comparativas têm em vista a construção de bases para formulação de metodologias que permitam a elaboração de mapas de quantidades através de processos automáticos. Os casos de estudo são evidenciados no Quadro 3.1.

Quadro 3.1- Análises comparativas entre diferentes modelos e versões IFC.

Análise Comparativa	Modelo Estrutural: IFC 4	Modelo Arquitetónico: IFC 2x3
Modelo Estrutural: IFC 2x3	X	X

A utilização de um modelo estrutural para IFC 4 é possível, uma vez que este foi construído manualmente e não através de uma ferramenta computacional. A sua proposta é realizada no capítulo 4. da Dissertação de Mestrado: “O Modelo IFC como Agente de Interoperabilidade” da autoria do Eng. Sérgio Pinho. Por sua vez, para o IFC 2x3 serão utilizadas as ferramentas computacionais *Tekla Structures*, *Autodesk Revit* e *Graphisoft ArchiCAD*.

O *Tekla Structures*, como o nome indica, é um programa direcionado exclusivamente para estruturas e, por isso, não opera com os módulos de arquitetura e de instalações. Já a última versão do *Autodesk Revit* (*Autodesk Revit 2014*) compreende todas estas vertentes. Os ficheiros que o programa da *Autodesk* gera são geralmente mais pesados dos que aqueles obtidos a partir do primeiro programa enunciado. Aliás, o *Tekla Structures* é considerado, pela generalidade dos utilizadores, uma ferramenta computacional com capacidades ao nível de detalhe e do modelo estrutural superiores ao *Autodesk Revit* [85]. É importante realçar que o *Autodesk Revit* ao incorporar diferentes projetos de especialidade num só programa, não

necessita diretamente do IFC para o seu desenvolvimento. Já o *Tekla Structures* pode depender da interoperabilidade entre ferramentas computacionais através do IFC para o desenvolvimento de projetos além do de estruturas.

Contudo, para o desenvolvimento do projeto de estruturas utilizando diferentes programas, como por exemplo os dois aqui referidos, poderá ser necessária a utilização do IFC. Porém, a transferência de informação entre programas por esta via poderá conduzir a erros, uma vez que o IFC ainda não está completamente estabelecido. Hoje em dia, face a este problema, existem ferramentas que permitem utilizar estes programas em simultâneo sem recorrer ao IFC. Uma destas formas é através do ISM (*Integrated Structure Modelling*) que permite obter uma boa gestão da mudança. Esta possibilidade é apresentada no subcapítulo 3.4.2.

Relativamente ao *Graphisoft ArchiCAD*, programa este que permite a conceção de modelos arquitetónicos e a respetiva orçamentação e planeamento, será utilizado para efetuar analogias com modelos arquitetónicos exportados a partir do *Autodesk Revit 2014*.

Para melhor perceção das possibilidades que estes *softwares* proporcionam, apresentam-se de seguida as opções iniciais que poderão ser selecionadas ao criar um novo projeto.

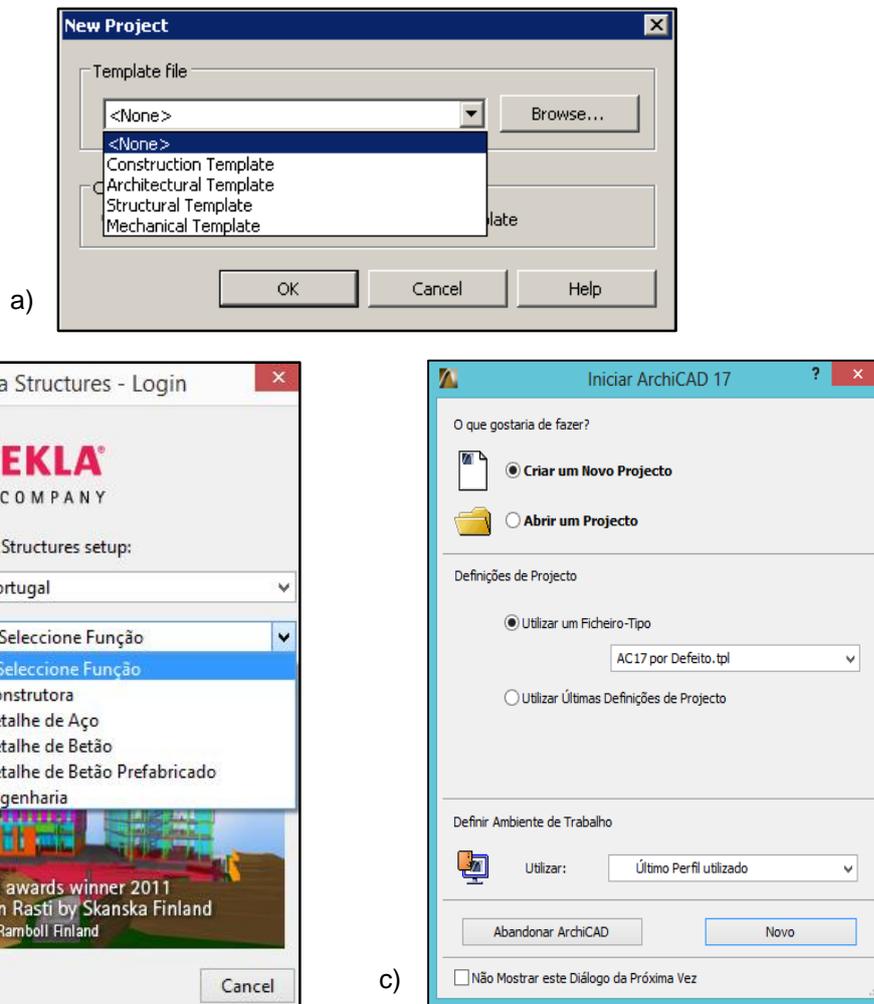


Figura 3.4 – Opções base: a) *Autodesk Revit*; b) *Tekla Structures*; c) *Graphisoft ArchiCAD*.

A escolha de uma destas opções influencia as definições que os programas utilizam por defeito. Por exemplo se a opção escolhida for “*Structural Template*” para o *Autodesk Revit*, então, é criada automaticamente uma vista com o modelo analítico, se não, esta terá que ser configurada através de processos manuais. Já para o *Tekla Structures* todas as opções base são referentes ao projeto de estruturas. Finalmente no *Graphisoft ArchiCAD* apenas existe uma opção para iniciar o projeto de arquitetura. As escolhas iniciais não podem voltar a ser alteradas no decorrer do projeto pela via da figura anterior, podendo, no entanto ser modificadas configurando o programa manualmente.

Quadro 3.2 - Softwares: Modelos Estruturais e Arquitetónicos.

Softwares	Modelo Estrutural	Modelo Arquitetónico
<i>Autodesk Revit</i>	SIM	SIM
<i>Tekla Structures</i>	SIM	NÃO
<i>Graphisoft ArchiCAD</i>	NÃO	SIM

3.4.2. ISM - INTEGRATED STRUCTURE MODELLING

O ISM (*Integrated Structure Modelling*) é um conceito de manipulação/ modificação de informação, utilizando normas estandardizadas, através da criação de uma base de dados ou repositório de informação associada a um modelo analítico de uma qualquer estrutura. Esta tecnologia é da autoria da Bentley, organização que apresenta soluções ao nível de ferramentas computacionais para áreas da construção civil.

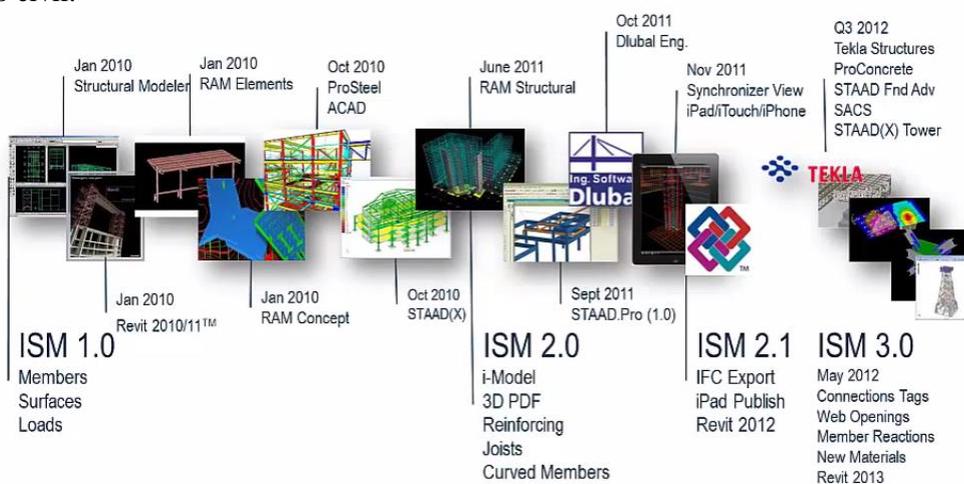


Figura 3.5 - Evolução do ISM. [86]

Este repositório pode ser gerado através de *softwares* da Bentley, como por exemplo, *RAM Structural System V8i*, *ProStructures V8i*, *AECOSim Building Designer*, entre outros. Contudo, pode também ser obtido através de ferramentas externas à Bentley, nomeadamente Tekla, *Revit Structures* e *Dlubal*, através da instalação de um complemento que se designa *Structural Synchronizer*. O repositório, depois de concebido com o formato “.ism.dgn.”, pode ser importado para qualquer um dos programas referidos e trabalhado a partir do seu ambiente de trabalho. [86]

Numa primeira fase é necessário efetuar uma sincronização de todas as famílias/ objetos utilizados. O conceito de família ou objeto é aplicado a qualquer elemento/ componente existente no modelo, sendo as componentes do mesmo tipo inseridas na família ou tipo de objeto correspondente. Estes elementos

podem ser definidos por variáveis, como por exemplo comprimento, largura e espessura, que por sua vez podem ser paramétricas ou não paramétricas. Depois de criados poderão, sempre que necessário, voltar a ser importados para um novo projeto. São exemplos de famílias/ objetos os pilares, as lajes, as vigas e as sapatas.

Após efetuarem-se todas as modificações no modelo, atualiza-se o repositório sendo possível selecionar quais as mudanças que se pretende que sejam efetivas. Esta é uma grande vantagem, na medida em que para além das alterações efetuadas pelo modelador, geram-se outras devidas à mudança de *software*. Por isso, ao ter o modelador no controle, não se introduzem modificações geradas por terceiros. A atualização pode ser acompanhada por um comentário descritivo do que foi executado.

Todas estas modificações são registadas e a cada atualização é indicado o historial de alterações efetuadas, isto é, que elementos foram alterados, adicionados e eliminados.

A Bentley apresenta também um software intitulado ISM Revision Manager a partir do qual é possível visualizar quem abriu um documento, quem o modificou e as descrições das alterações realizadas. Além disso, permite comparar diferentes atualizações e ainda recuperar qualquer dado perdido. Hoje em dia a existência de um cadastro composto com todas estas informações é fundamental para uma boa gestão de projetos. Na Figura 3.6, apresenta-se um resumo da aplicabilidade da ISM.

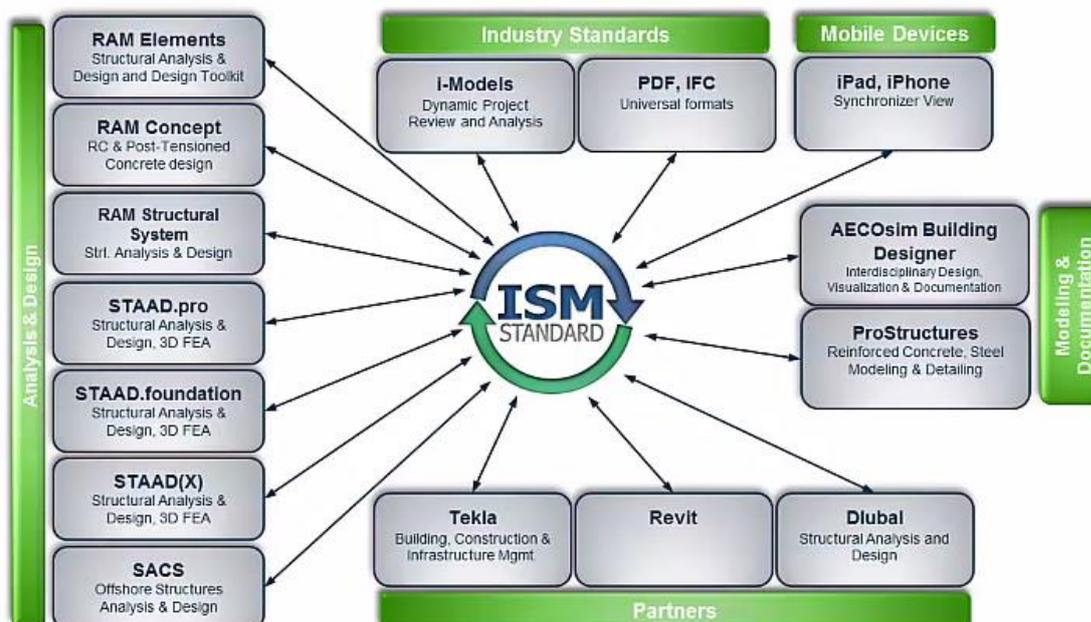


Figura 3.6 – Aplicabilidade da ISM [87]

Se a tendência de implementação desta ferramenta continuar ao ritmo atual, então, poderá ser possível a sua aplicação à escala mundial para trocas de informação ao nível do modelo analítico, de modo a executarem-se as análises estruturais necessárias. Não se quer com isto dizer, que o papel do IFC será menos pertinente. Claro que a aplicação desta ferramenta para a troca de informação entre diferentes programas terá vantagens em relação ao IFC, uma vez que, por exemplo, permite a escolha da informação que pretendemos adicionar ao atualizar o repositório existente. No entanto, este processo constitui apenas uma pequena fração de todo um ciclo de vida de um projeto que pode ser obtido através o IFC. Como se pode observar pela Figura 3.5 e pela Figura 3.6, a última versão do ISM possibilita a exportação para IFC. Esta realidade, juntamente com a possibilidade de controlar a informação que é adicionada ao repositório, permite a criação de modelos IFC mais fidedignos.

3.5. MODELOS ANALÍTICOS PARA IFC 2x3 E IFC 4

Neste subcapítulo, são analisados modelos estruturais em IFC 2x3 concebidos a partir do *Autodesk Revit* e *Tekla Structures* e o modelo estrutural em IFC 4, concebido manualmente. Este último, como não foi gerado por uma ferramenta computacional, apresenta com uma boa organização e é composto apenas pelas informações estritamente necessárias, sendo a sua representação gráfica exposta na Figura 3.7.

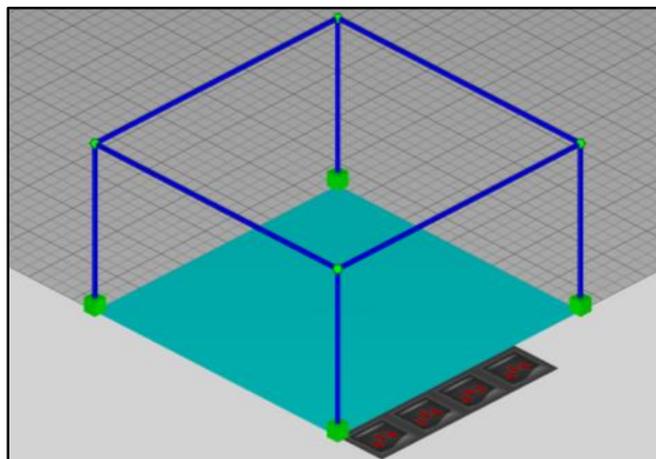


Figura 3.7 – Modelo Analítico IFC 4 [68]

Os elementos estruturais que compõem esta estrutura são pilares, vigas, laje maciça e sapatas isoladas. Este modelo constitui um bom ponto de partida, não só para evidenciar as diferenças no modo como a informação pode ser estruturada (comparativamente à sua conceção automática através de uma ferramenta computacional), mas também para estabelecer comparações entre as versões referidas. Apesar de neste modelo em IFC 4 não ter sido possível a atribuição de uma cota para a laje da estrutura, considera-se relevante a sua utilização, uma vez que num futuro próximo esta versão será a mais predominante.

Para efetuar uma comparação com o IFC 2x3 será concebido um modelo analítico com uma tipologia estrutural semelhante à da Figura 3.7 a partir das duas ferramentas computacionais referidas, diferindo apenas nas dimensões dos elementos e na cota da laje, como é apresentado na Figura 3.8. Tendo em conta que esta diferença de cotas é uma informação adquirida, então, quaisquer diferenças originadas por esta, poderão ser facilmente constatadas. Sempre que alguma diferença na estrutura do código seja devida à diferença de versões, esta será oportunamente exposta.

Refere-se ainda, que na formulação de modelos BIM é fundamental a utilização de famílias/ objetos, para conseqüente exportação para IFC. Por outras palavras, e exemplificando, o aspeto físico de um pilar pode ser obtido através de um conjunto de linhas traçadas diretamente no modelo. Se assim for, a ferramenta computacional lê cada linha singularmente e não o conjunto de linhas que formam o elemento. Neste sentido, após exportação do modelo para IFC, não existirá nenhuma informação no ficheiro que remeta para o pilar criado, pois na realidade existe apenas um conjunto de linhas. Esta situação é solucionada empregando famílias/ objetos existentes ou concebendo-os de raiz. Para conceber um elemento original, modela-se a estrutura em 3D e escolhe-se a família/ objeto a que este pertence. Por exemplo, após modelar um pilar e indicar ao programa que a estrutura concebida é um pilar, o programa associa uma entidade IFC ao elemento criado, sendo esta, posteriormente, incluída no ficheiro IFC. É, portanto, necessário modelar com valor semântico. Através deste raciocínio é facilmente perceptível que a formulação de modelos BIM é muito mais do que uma simples modelação 2D ou 3D.

Os elementos modelados podem ser paramétricos, ou seja, ao alterar uma única dimensão, todas as outras se alteram de um modo proporcional ao objeto inicial.

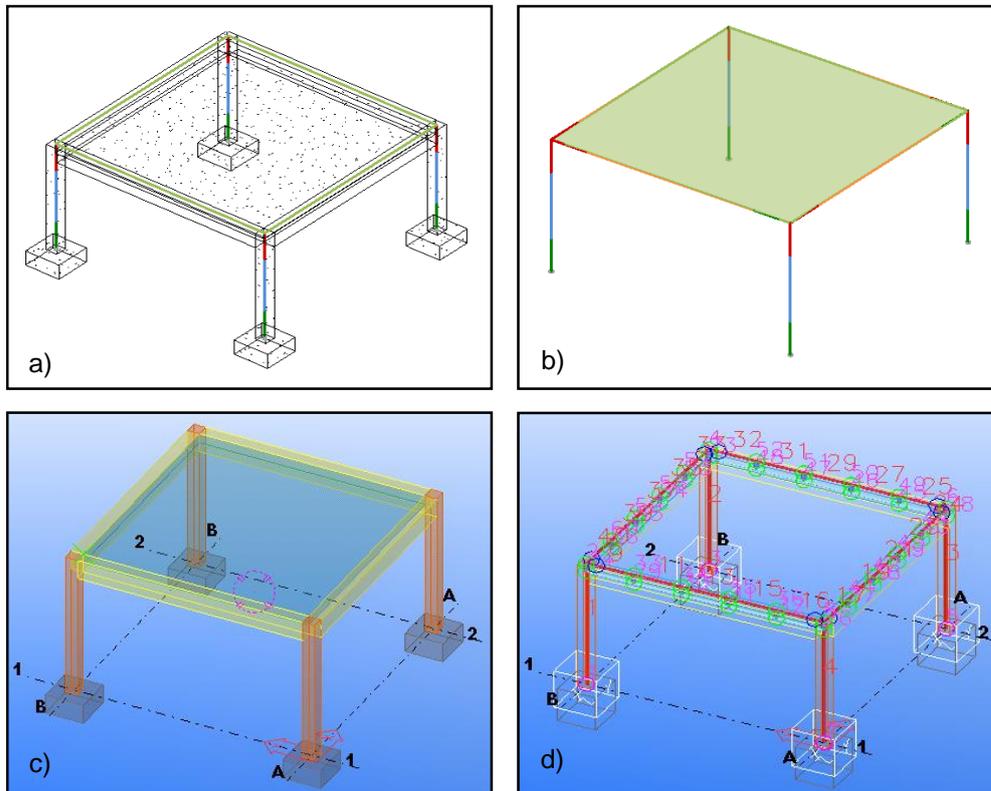


Figura 3.8 – Autodesk Revit: a) Modelo 3D; b) Modelo Analítico. Tekla Structures: c) Modelo 3D; b) Modelo Analítico (IFC 2x3).

3.5.1. IDENTIFICAÇÃO DE MATERIAIS

O primeiro objetivo passa por determinar qual ou quais as entidades que permitem identificar os materiais empregues numa determinada obra. Conclui-se que a entidade onde são armazenadas estas informações denomina-se “*IfcMaterial*”. Na Figura 3.9 apresentam-se quais os atributos que a compõem, assim como exemplos da sua aplicação para ambas as versões IFC e ferramentas computacionais em estudo.

IDENTIFICAÇÃO DO MATERIAL	(IFC 2x3 genérico)
#n= IFCMATERIAL (“Nome”, “Descrição”, “Categoria”);	
- Autodesk Revit	(IFC 2x3 exemplo)
#120= IFCMATERIAL (“Concrete - Cast-in-Place Concrete”);	
- Tekla Structures	(IFC 2x3 exemplo)
#345= IFCMATERIAL('CONCRETE/C45/55')	
IDENTIFICAÇÃO DO MATERIAL	(IFC 4 genérico)
#n= IFCMATERIAL (“Nome”, “Descrição”, “Categoria”);	
- Conceção Manual	(IFC 4 exemplo)
#700= IFCMATERIAL ('C20/25', \$, 'Concrete');	

Figura 3.9 – Identificação do Material: “*IfcMaterial*”.

Portanto, as diferenças entre ambas as versões cingem-se ao número de atributos que esta entidade utiliza. Enquanto no IFC 2x3 apenas é utilizado o atributo “Nome”, no IFC 4 os atributos abrangidos são “Nome”, “Descrição” e “Categoria”. Como no IFC 2x3 todas as informações referentes ao material utilizado são introduzidas num único atributo, então, podem gerar-se designações confusas.

Nos exemplos apresentados para o IFC 2x3, o nome do material exposto é definido a partir do ambiente de trabalho do programa, nomeadamente “*Concrete - Cast-in-Place Concrete*” para o *Autodesk Revit* e “*CONCRETE/C45/55*” para o *Tekla Structures*. Estas designações encontram-se definidas por defeito e, por isso, denota-se uma maior especificação nas bibliotecas do *software* direcionado exclusivamente para estruturas. Contudo, estas informações podem ser definidas pelo modelador, alterando as propriedades do material no ambiente de trabalho da ferramenta computacional. Uma vez que existe apenas um atributo nesta entidade, as informações nele contidas terão que ser alvo de rigor por parte do modelador.

Relativamente ao IFC 4, o campo “Nome” é menos generalista que o campo “Categoria” e como a designação indica, em “Descrição” utilizam-se termos mais descritivos. Neste exemplo, pode comprovar-se a diferença referida entre “Nome” e “Categoria” não tendo, contudo, sido definido o conteúdo para “Descrição”. Por exemplo, a *buildingSMART* aconselha a utilização de expressões como “*Steel*” ou “*Concrete*” para o atributo “Categoria”, como foi aqui executado [88].

Como foi anteriormente referido, a única ferramenta que opera com o IFC 4 denomina-se *Construtivity* e, por este motivo, este formato não é ainda importável para os restantes programas. Contudo, no futuro a realidade será outra e, conseqüentemente, todas as informações contidas nos atributos da entidade “*IfcMaterial*” serão gerados automaticamente e não manualmente, como aqui exposto. Neste sentido, o papel que os programas desempenharão, na criação das bibliotecas de materiais e na adaptação das respetivas nomenclaturas para estes atributos, será de extrema importância para a identificação conjunta de materiais empregues. Evidentemente, estes atributos poderão, também, ser definidos manualmente pelo modelador através do ambiente de trabalho do programa, especificando as nomenclaturas pretendidas.

Em virtude dos fatos mencionados, existe uma tendência clara de evolução entre as versões analisadas, no sentido de uma melhor divisão de conteúdos.

3.5.2. IDENTIFICAÇÃO DE ELEMENTOS

A cada elemento estrutural corresponde uma entidade principal, podendo existir ou não supra entidades destas entidades principais. As supra entidades são, geralmente, utilizadas para identificar diferentes tipos de elementos estruturais existentes. Por outras palavras, uma entidade deste tipo representa um conjunto de elementos com características semelhantes. Apresentam-se as entidades presentes nos exemplos em análise no Quadro 3.3.

Quadro 3.3 - Identificação de elementos estruturais (Modelos Estruturais).

ELEMENTOS	ENTIDADES PRINCIPAIS			SUPRAENTIDADES		
	IFC 2x3	IFC 2x3	IFC 4	IFC 2x3	IFC 2x3	IFC 4
	<i>Tekla Structures</i>	<i>Autodesk Revit</i>	Conceção Manual	<i>Tekla Structures</i>	<i>Autodesk Revit</i>	Conceção Manual
Pilares	<i>IfcColumn</i>	<i>IfcColumn</i>	<i>IfcColumn</i>	<i>IfcColumnType</i>	<i>IfcColumnType</i>	-
Vigas	<i>IfcBeam</i>	<i>IfcBeam</i>	<i>IfcBeam</i>	<i>IfcBeamType</i>	-	-
Laje	<i>IfcSlab</i>	<i>IfcSlab</i>	<i>IfcSlab</i>	<i>IfcSlabType</i>	-	-
Sapatas	<i>IfcFooting</i>	<i>IfcFooting</i>	<i>IfcFooting</i>	-	-	-

Como se pode inferir pelo Quadro 3.3, as entidades principais repetem-se para os ficheiros em análise, independentemente do programa pelo qual se opera ou versão IFC. Quanto às supra entidades, pode constatar-se uma inconstância da sua presença de programa para programa. O modo de representação das entidades principais é exposto na Figura 3.10 e das supraentidades na Figura 3.11.

```
IDENTIFICAÇÃO DE ELEMENTOS ESTRUTURAIS (IFC 2x3 e IFC 4 genérico)
#n= IFCCOLUMN; IFCBEAM; IFCSLAB; IFCFOOTING
  ("GUID", "Historial do usuário", "Nome", "Descrição", "Tipo do objeto",
  "Posicionamento do objeto", "Representação", "Etiqueta", "Tipo pré-definido");
( IFC 2x3 e IFC 4 exemplo)
#259= IFCCOLUMN ('18KVJfJGL26B_mTRg$Y$Zc', #41, 'Concrete-Square-Column(1):25 x 25:
158880', $, '25 x 25', #257, #250, '158880');
```

Figura 3.10 – “*IfcColumn*”; “*IfcBeam*”; “*IfcSlab*”; “*IfcFooting*”.

```
IDENTIFICAÇÃO DE TIPOS DE ELEMENTOS ESTRUTURAIS (IFC 2x3 e IFC 4 genérico)
#n = IFCCOLUMNNTYPE; IFCBEAMNTYPE; IFCSLABNTYPE; IFCFOOTINGNTYPE3
  ("GUID", "Historial do usuário", "Nome", "Descrição", "Ocorrência Aplicável",
  "Tem conjunto de propriedades", "Mapa de representação", "Etiqueta", "Tipo de
  elemento", "Tipo pré-definido")
( IFC 2x3 e IFC 4 exemplo)
#117= IFCCOLUMNNTYPE ('18KVJfJGL26B_mTRg$Y$c1', #41, '25 x 25', $, $, $, (#115), '159209',
'25 x 25', .COLUMN.);
```

Figura 3.11 – “*IfcColumnType*”; “*IfcBeamType*”; “*IfcSlabType*”; “*IfcFootingType*”.

Como se constata pela figura anterior, à exceção da supraentidade “*IfcFootingType*”, que existe apenas para a versão IFC 4, todas as outras apresentadas são comuns a ambas as versões e apresentam o mesmo modo de representação.

³ “*IfcFootingType*” é uma entidade existente apenas para a versão IFC 4.

Desta forma, como a presença das entidades com a terminologia “Type” não é constante, então, o modo de identificação destes elementos deve ser realizado segundo as entidades apresentadas na Figura 3.10.

3.5.3. ASSOCIAÇÃO DE MATERIAIS A ELEMENTOS

Após a identificação dos materiais e dos elementos estruturais presentes é necessário determinar o modo como as relações entre estes são estabelecidas. O IFC organiza toda esta informação através de uma entidade denominada “IfcRelAssociatesMaterial” (Figura 3.12).



Figura 3.12 – Relação estabelecida por “IfcRelAssociatesMaterial”.

Os atributos pelos quais esta entidade é composta são apresentados na Figura 3.13, assim como um exemplo prático da sua aplicação, o qual é aplicável a qualquer versão IFC em estudo e ferramenta computacional.

ASSOCIAÇÃO DE MATERIAIS A ELEMENTOS	(IFC 2x3 e IFC 4 genérico)
#n= IFCREASSOCIATESMATERIAL("\`GUID' ", "Historial do usuário", "Nome", "Descrição", "Objetos a relacionar", "Material relacionado");	
	(IFC 2x3 e IFC 4 exemplo)
#906= IFCREASSOCIATESMATERIAL('2RJq8t\$Rr30uIdhOlcdwn', #5, \$, \$, (#227, #179, #131, #82), #113);	

Figura 3.13 - “IfcRelAssociatesMaterial” (IFC 4 e IFC 2x3).

No primeiro atributo pode encontrar-se um código do tipo '2RJq8t\$Rr30uIdhOlcdwn', ou seja, o *GUID* (Global Unique Identifier) composto por 22 caracteres aleatórios, que servem para identificação de objetos. Existem, também, outras formas de identificação, como o “Nome” e “Descrição”, contudo a vantagem do GUID passa por ser sempre o mesmo em quaisquer trocas de informações entre ferramentas computacionais.

De seguida, no atributo “Objetos a relacionar”, são apresentadas todas as entidades que representam elementos com o mesmo material. Para a identificação desse material utiliza-se o último atributo, nomeadamente “Material relacionado”. É através destes dois últimos atributos que as relações entre materiais e elementos são estabelecidas. Contudo, a forma como estas são estabelecidas varia consoante a versão IFC e ferramenta computacional em análise.

Neste sentido, analisaram-se os casos práticos em estudo e exemplifica-se, de seguida, quais os tipos de relações que podem ser estabelecidas a partir desta entidade e quais os resultados obtidos.

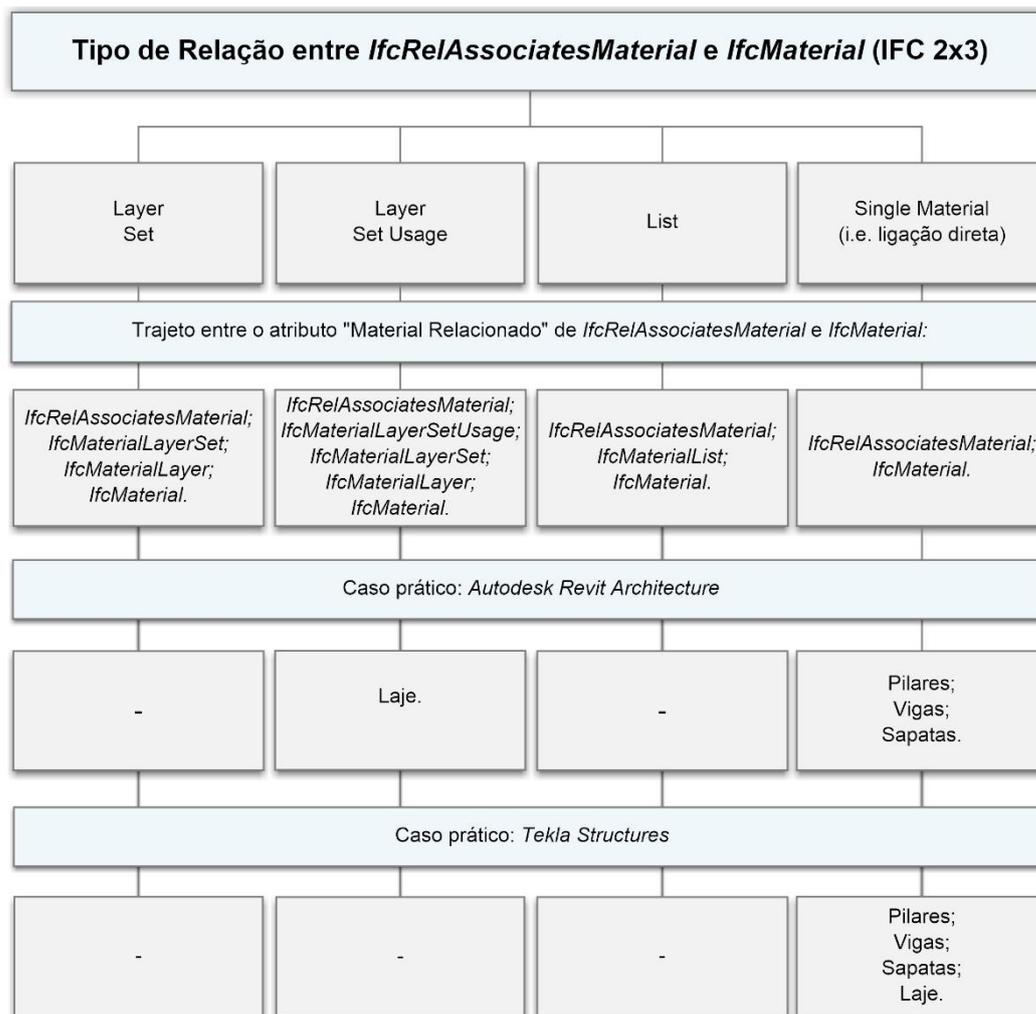


Figura 3.14 – Relações entre “*IfcRelAssociatesMaterial*” e “*IfcMaterial*” (IFC 2x3).

Pela análise dos casos práticos formulados a partir do *Autodesk Revit* e *Tekla Structures*, ou seja, para o IFC 2x3, constata-se que as relações estabelecidas entre “*IfcRelAssociatesMaterial*” e “*IfcMaterial*” são idênticas, à exceção daquela instituída para o elemento estrutural: Laje. Apesar de apresentarem relações diferentes para este elemento, o material é reconhecido após importação dos ficheiros IFC em estudo para uma ferramenta BIM.

Os elementos podem ser articulados a um material através de uma relação do tipo “*Single Material*” sempre que a sua definição não o proíba a tal. Para esta versão IFC não são apresentadas recomendações por parte da *buildingSMART* para o modo como estas relações devem ser estabelecidas, ao contrário do que sucede para o IFC 4 [89].

De seguida, apresenta-se a relação constatada para a laje, a partir dos dois *softwares* enunciados, de modo a obter-se uma melhor perceção do que foi atrás indicado.

```

RELAÇÃO ENTRE IFCREASSOCIATESMATERIAL E IFCMATERIAL: LAJE      Autodesk Revit
#1192= IFCREASSOCIATESMATERIAL('31Qji30qr4B8B1BiTIpP9Q',#41,$,$, (#1031),#1084);
#1084= IFCMATERIALLAYERSETUSAGE(#1081,.AXIS3.,.POSITIVE.,0.);
#1081= IFCMATERIALLAYERSET((#1079),'Foundation Slab:30');
#1079= IFCMATERIALLAYER(#512,0.3,$);
#512= IFCMATERIAL('Concrete - Cast In Situ - Bet\X2\00E3\X0\o');
#1031= IFCSLAB('18KVJfJGL26B_mTRg$Y_Hx',#41,'Foundation Slab:30:159805',$,
'Foundation Slab:30',#1015,#1029,'159805',.FLOOR.);

RELAÇÃO ENTRE IFCREASSOCIATESMATERIAL E IFCMATERIAL: LAJE      Tekla Structures
#907= IFCREASSOCIATESMATERIAL('2Q1$oy1C5BwBbQ1roNxSkj',#5,$,$, (#772,#691,#642,
#594,#540,#458,#411,#363,#308),#345);
#345= IFCMATERIAL('CONCRETE/C45/55');
#772= IFCSLAB('1JHcGg0000D34pEJSnCZ8o',#5,'LAJE BETAO','200*4750','200*4750',
#736,#771,'ID5346642a-0000-0034-3133-393731323232',.FLOOR.);
    
```

Figura 3.15 - Relação entre “IfcRelAssociatesMaterial” e “IfcMaterial” para Laje (casos de estudo).

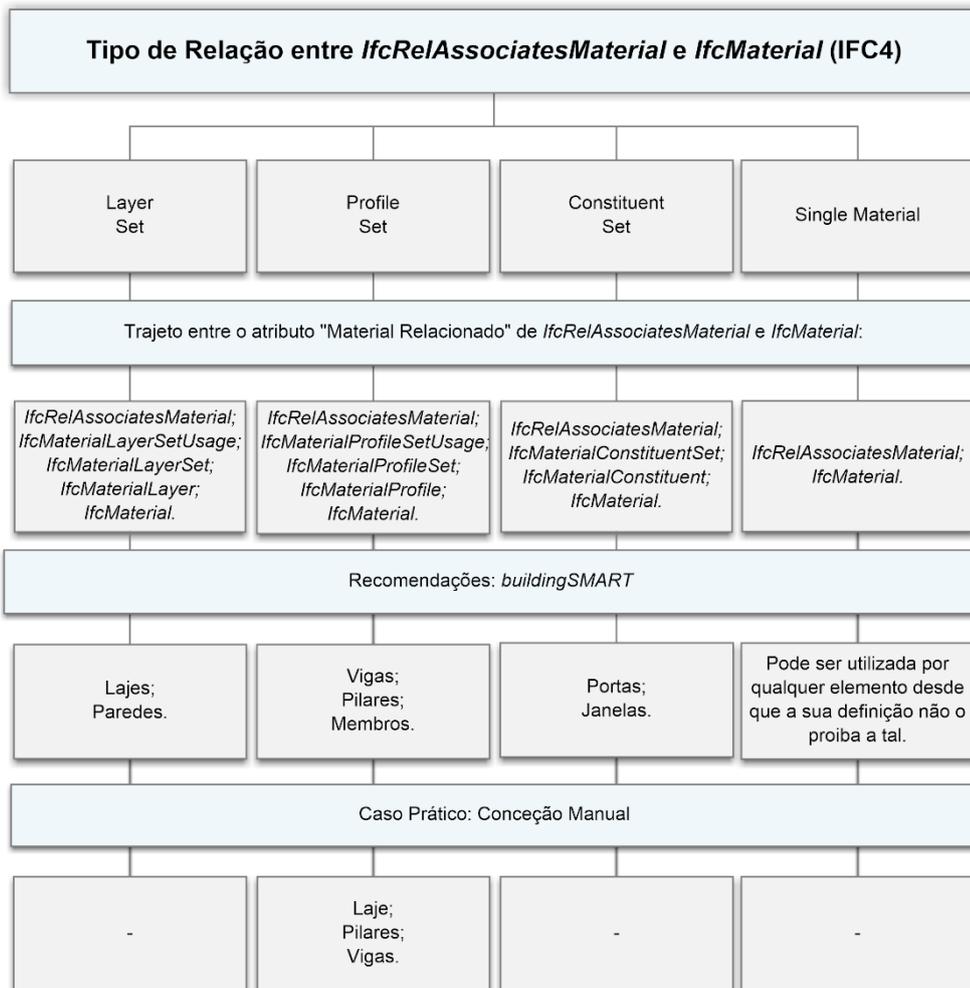


Figura 3.16 – Relações entre “IfcRelAssociatesMaterial” e “IfcMaterial” (IFC 4).

Do IFC 2x3 para o IFC 4, enquanto as relações do tipo “*Layer Set Usage*” e “*List*” são excluídas, as relações “*Profile Set*” e “*Constituent Set*” são adicionadas.

A *buildingSMART* recomenda, para o IFC 4, a organização da informação segundo os critérios definidos na figura anterior, no entanto como se pode observar para o caso prático em estudo, estas relações não foram estabelecidas como recomendado para a laje existente. Todavia, o material é reconhecido através do *Construtivity*. Relativamente às sapatas, não lhes é atribuída nenhuma relação uma vez que estas não foram associadas a um material.

Por estas razões, conclui-se que o modo como as relações se estabelecem não é de carácter obrigatório, tanto para o IFC 2x3 como para o IFC 4, podendo estas variar mediante as configurações de cada ferramenta programática.

3.5.4. UNIDADES GEOMÉTRICAS

Antes de proceder-se à determinação das informações geométricas é necessário investigar qual a unidade adotada no projeto.

As unidades de trabalho podem ser definidas durante todo o decorrer do projeto, sendo contudo aconselhado a sua escolha no início. Apesar desta possibilidade, não existe uma opção direta que permita escolher as unidades no momento de exportação do modelo para IFC, como se observa pela Figura 3.17.

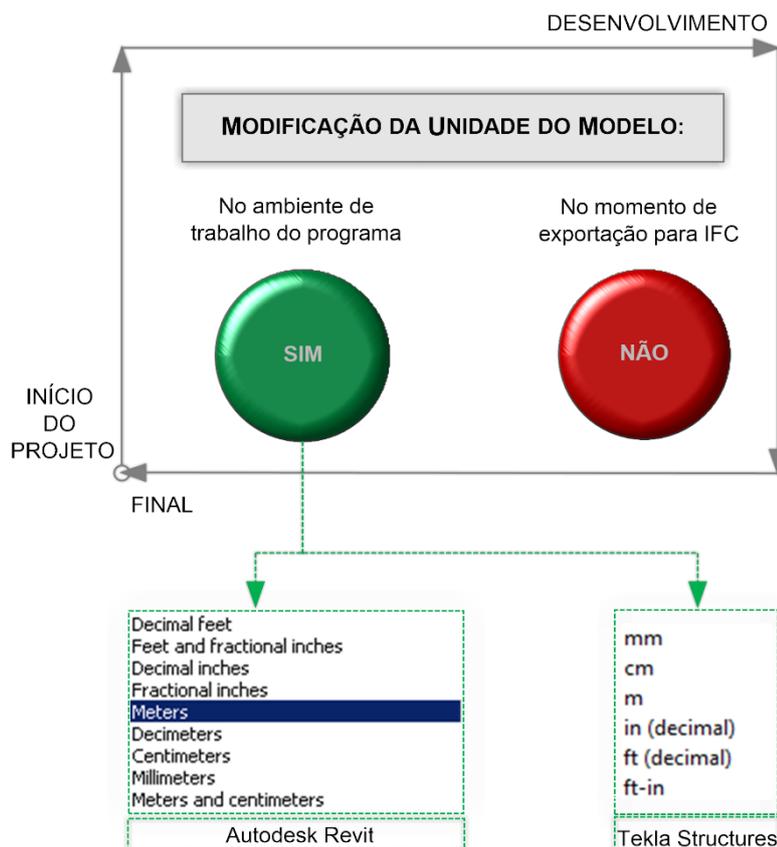


Figura 3.17 – Modificação das Unidades no Modelo: Autodesk Revit e Tekla Structures (IFC 2x3).

Para a sua determinação através do modelo gerado em IFC é necessário estudar a entidade que contém estas informações, a qual denomina-se “*IfcSIUnit*”. Na Figura 3.18, apresenta-se a sua representação genérica, assim como exemplos práticos da sua aplicação com diferentes unidades de comprimento.

IDENTIFICAÇÃO DA UNIDADE		(IFC 4 e IFC2x3 genérico)
#n= IFCSIUNIT (*, "Tipo de Unidade", "Prefixo", "Nome");		
Exemplos para Metros		
IFC 2x3		
Tekla Structures:	#n= IFCSIUNIT (*, .LENGTHUNIT., .MILLI., .METRE.);	
Autodesk Revit:	#n= IFCSIUNIT (*, .LENGTHUNIT., \$, .METRE.);	
IFC 4		
Conceção Manual:	#n= IFCSIUNIT (*, .LENGTHUNIT., \$, .METRE.);	
Exemplos para Decímetros		
IFC 2x3		
- Tekla Structures:	#n= IFCSIUNIT (*, .LENGTHUNIT., .MILLI., .METRE.);	
- Autodesk Revit:	Não está implementado corretamente.	
IFC 4		
Conceção Manual:	#n= IFCSIUNIT (*, .LENGTHUNIT., .DECI., .METRE.);	
Exemplos para Centímetros		
IFC 2x3		
- Tekla Structures:	#n= IFCSIUNIT (*, .LENGTHUNIT., .MILLI., .METRE.);	
- Autodesk Revit:	#n= IFCSIUNIT (*, .LENGTHUNIT., .CENTI., .METRE.);	
	#n+1= IFCSIUNIT (*, .LENGTHUNIT., \$, .METRE.);	
IFC 4		
- Conceção Manual:	#n= IFCSIUNIT (*, .LENGTHUNIT., .CENTI., .METRE.);	
Exemplos para Milímetros		
IFC 2x3		
- Tekla Structures:	#n= IFCSIUNIT (*, .LENGTHUNIT., .MILLI., .METRE.);	
- Autodesk Revit:	#n= IFCSIUNIT (*, .LENGTHUNIT., .MILLI., .METRE.);	
	#n+1= IFCSIUNIT (*, .LENGTHUNIT., \$, .METRE.);	
IFC 4		
- Conceção Manual:	#n= IFCSIUNIT (*, .LENGTHUNIT., .MILLI., .METRE.);	

Figura 3.18 - “*IfcSIUnit*”.

Para distâncias entre dois pontos esta entidade utiliza no atributo “Tipo de Unidade” a designação “*LENGTHUNIT.*”. No atributo “Prefixo” é estabelecida a unidade empregue e, por fim, no atributo “Nome” é apresentada a unidade no Sistema Internacional. Esta entidade pode também ser utilizada como representação de outros tipos de unidade de medição, como se apresenta na Figura 3.19.

```

DIFERENTES TIPOS DE UNIDADES DE MEDIÇÃO (exemplos)
#n= IFCSIUNIT(*, .LENGTHUNIT., $, .METRE.);
#n= IFCSIUNIT(*, .AREAUNIT., $, .SQUARE_METRE.);
#n= IFCSIUNIT(*, .VOLUMEUNIT., $, .CUBIC_METRE.);
#n= IFCSIUNIT(*, .PLANEANGLEUNIT., $, .RADIAN.);
#n= IFCSIUNIT(*, .MASSUNIT., .KILO., .GRAM.);
#n= IFCSIUNIT(*, .TIMEUNIT., $, .SECOND.);
#n= IFCSIUNIT(*, .THERMODYNAMICTEMPERATUREUNIT., $, .KELVIN.);
#n= IFCSIUNIT(*, .POWERUNIT., $, .WATT.);

```

Figura 3.19 - “IfcSIUnit”: Diferentes tipos de unidades de medição.

Relativamente a estas unidades será, também, importante ter em conta as unidades de volume e de área, uma vez que existe a possibilidade de existirem informações geométricas associadas a estas unidades. Cada tipo de unidade é independente de outra existente e, por isso, é possível existirem simultaneamente informações no modelo IFC em m, cm² e mm³.

Voltando ao exemplo em análise para “.LENGTHUNIT.” e IFC 4, tem-se que, sempre que o conteúdo do atributo “Prefixo” for constituído pelo símbolo “\$” a unidade adotada é, por defeito, a do Sistema Internacional, ou seja, metros. Se este atributo for constituído pela designação “.DECL.”, “.CENTI.” ou “.MILLI” as unidades são decímetros, centímetros e milímetros, respetivamente.

Para o IFC 2x3, a situação verificada é um pouco diferente e depende do *software* em análise. Para o *Autodesk Revit*, existe sempre uma entidade “IfcSIUnit” com a tipologia apresentada para a unidade metros. Neste sentido, pelo raciocínio anterior, automaticamente se inferia que a unidade adotada seria metros. No entanto, tal não é totalmente fidedigno. A unidade utilizada será metros se apenas existir uma única entidade “IfcSIUnit” com a tipologia referida. Caso a unidade empregue for centímetros ou milímetros, então, será gerada uma outra entidade “IfcSIUnit” com o “Prefixo” definido por .CENTI. para centímetros ou .MILLI. para milímetros.

Seria de esperar uma metodologia semelhante para a unidade decímetros, no entanto, ao executarem-se testes no *Autodesk Revit* adotando esta unidade, o modelo IFC obtido apenas apresenta a entidade “IfcSIUnit” com a tipologia correspondente à unidade metros. Neste sentido, foi feita uma importação do modelo em decímetros para o ambiente de trabalho desta ferramenta programática via IFC, de modo a determinar qual a unidade indicada pelo próprio *software*. A unidade obtida foi “Feet and fractional inches”. No entanto, enquanto as propriedades geométricas dos pilares são convertidas para a unidade decímetros, as propriedades dos restantes elementos são modificadas para valores incorretos e não correspondentes a qualquer tipo de unidade.

Esta situação revela a existência de um possível erro na conversão do modelo. De modo a aprofundar a investigação, foi utilizado o *Solibri*, um visualizador de modelos IFC, tendo-se, alcançado o mesmo resultado. Este modelo, em decímetros, importado para estas duas ferramentas é apresentado na Figura 3.20.

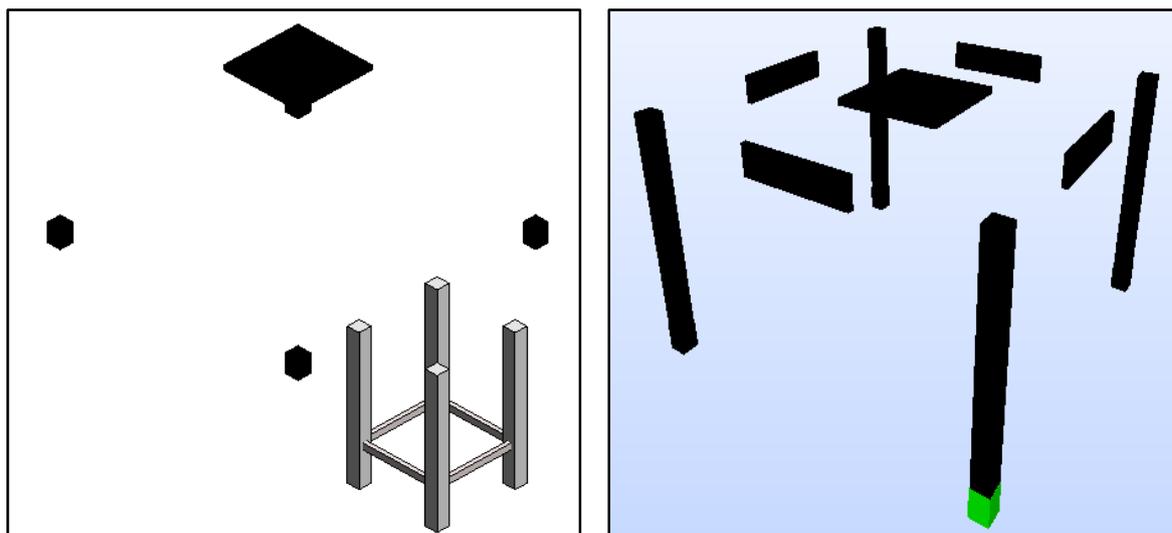


Figura 3.20 - Modelo em decímetros: a) Autodesk Revit; b) Solibri.

O problema de existirem duas entidades “*IfcSIUnit*” para unidades de comprimento pode ser facilmente ultrapassado através da entidade “*IfcUnitAssignment*”. Nesta entidade é feita referência a todas as unidades adotadas no projeto, sendo que para unidades de comprimento apenas é encontrada uma única referência à entidade “*IfcSIUnits*”. A título de exemplo, apresenta-se um exemplo prático para a unidade centímetros através do Autodesk Revit.

```
Autodesk Revit - IFC 2x3
Exemplo aplicado a centímetros
#63= IFCUNITASSIGNMENT( (#42, #44, #45, #49, #50, #51, #52, #56, #60, #62) );
#42= IFCSIUNIT( *, .LENGTHUNIT., .CENTI., .METRE. );
#43= IFCSIUNIT( *, .LENGTHUNIT., $, .METRE. );
```

Figura 3.21 - Autodesk Revit. Exemplo aplicado a centímetros.

Relativamente às unidades no *Tekla Structures* a situação é um pouco diferente. Qualquer ficheiro obtido a partir deste programa, utilizando uma qualquer unidade, apresenta uma entidade “*IfcSIUnit*” aplicada à unidade milímetros sendo, por isso, esta a unidade adotada em qualquer exportação, independentemente da unidade de projeto. Apesar de existir uma impossibilidade de atribuição de uma unidade diferente de milímetros no modelo em IFC, testes realizados no *Solibri*, não revelam qualquer erro nos ficheiros concebidos.

Refere-se, ainda que, o *Tekla Structures* não utiliza importação direta via IFC, mas através de uma opção intitulada “*Insert Reference Model...*”. Esta opção permite importar diferentes tipos de formatos como, por exemplo, *.dxf;*.dwg;*.dgn;*.ifc.;*.ifcZIP; *.ifcXML. Ao fazer importação por esta via há ainda que selecionar qual a relação existente entre a escala do ficheiro e a escala do ambiente de trabalho do programa.

3.5.5. DIMENSÕES GEOMÉTRICAS

Após a identificação das relações que tornam possível a associação de elementos a materiais e identificação das unidades geométricas, procede-se para a determinação das informações relativas à

geometria desses mesmos elementos, de modo a contabilizar as quantidades. Para concretizar esta tarefa serão analisados em paralelo os modelos para IFC 2x3 e IFC 4, comparando elementos da mesma família.

Primeiramente, apresenta-se uma comparação entre caso de estudo aplicado ao *Autodesk Revit* (IFC 2x3) e o concebido manualmente (IFC 4). Posteriormente, apresenta-se esta análise aplicada ao *Tekla Structures*, pois o modo como este último opera é no seu todo diferente.

RELAÇÕES PARA OBTEN AS PROPRIEDADES GEOMÉTRICAS: PILARES	(IFC 2x3 exemplo)
#1195= IFCRELAASSOCIATESMATERIAL ('3kz6MA6Vb4kxyQFp7ojR7F', #41, \$, \$, (#117, #259, #345, #397, #449), #120);	
#117= IFCCOLUMNTYPE ('18KVJfJGL26B_mTRg\$Y\$cl', #41, '25 x 25', \$, \$, \$, (#115), '159209', '25 x 25', .COLUMN.);	
#115= IFCREPRESENTATIONMAP (#114, #109);	
#109= IFCSHAPEREPRESENTATION (#72, 'Body', 'SweptSolid', (#108));	
#108= IFCEXTRUDEDAREASOLID (#106, #107, #19, 3.);	
#106= IFCRECTANGLEPROFILEDEF (.AREA., \$, #105, 0.25, 0.25);	
#259= IFCCOLUMN ('18KVJfJGL26B_mTRg\$Y\$Zc', #41, 'Concrete-Square-Column(1):25 x 25:158880', \$, '25 x 25', #257, #250, '158880');	
#250= IFCPRODUCTDEFINITIONSHAPE (\$, \$, (#248));	
#248= IFCSHAPEREPRESENTATION (#72, 'Body', 'MappedRepresentation', (#246));	
#246= IFCMAPPEDITEM (#115, #245);	
#115= IFCREPRESENTATIONMAP (#114, #109);	
#109= IFCSHAPEREPRESENTATION (#72, 'Body', 'SweptSolid', (#108));	
#108= IFCEXTRUDEDAREASOLID (#106, #107, #19, 3.);	
#106= IFCRECTANGLEPROFILEDEF (.AREA., \$, #105, 0.25, 0.25);	
(idem para #345, #397 e #449)	
RELAÇÕES PARA OBTEN AS PROPRIEDADES GEOMÉTRICAS: PILARES	(IFC 4 exemplo)
#719= IFCRELAASSOCIATESMATERIAL ('01kvXT9hb7C80NqRflZZF9', #5, \$, \$, \$, (#301, #311, #321, #331, #439, #444, #449, #454), #718);	
#439= IFCCOLUMN ('1fv4DZfy55exwX8QDy8xmw', #5, 'Column1', \$, \$, #438, #429, \$, .COLUMN.);	
#429= IFCPRODUCTDEFINITIONSHAPE (\$, \$, (#427));	
#427= IFCSHAPEREPRESENTATION (#202, 'Body', 'SweptSolid', (#426));	
#426= IFCEXTRUDEDAREASOLID (#425, #191, #13, 3.0);	
#425= IFCRECTANGLEPROFILEDEF (.AREA., '20x30', \$, 0.20, 0.30);	
(idem para #444, #449 e #454)	

Figura 3.22 – Relações para obter as propriedades geométricas dos Pilares para os modelos: *Autodesk Revit* (IFC 2x3) e Concebido Manualmente (IFC 4).

Portanto, as classes onde são definidas as propriedades geométricas de pilares nos casos em estudo indicados são “*IfcExtrudedAreaSolid*” e “*IfcRectangleProfileDef*”. Note que caso existam pilares de secção circular a entidade apresentada será “*IfcCircleProfileDef*” em vez de “*IfcRectangleProfileDef*”. A principal diferença consiste no modo como estas entidades são obtidas.

ENTIDADES QUE CONTÊM AS PROPRIEDADES GEOMÉTRICAS	(IFC 2x3 e IFC 4 genérico)
#n= IFCEXTRUDEDAREASOLID ("Swept Área", "Posição", "Direção da extrusão", "Profundidade");	
#n= IFCRECTANGLEPROFILEDEF ("Tipo de perfil", "Nome do perfil", "Posição", "Dimensão X", "Dimensão Y");	

Figura 3.23 – Entidades que contêm as propriedades geométricas: “*IfcExtrudedAreaSolid*” e “*IfcRectangleProfileDef*” (IFC 4 e IFC 2x3).

A partir destas três dimensões, nomeadamente “Profundidade”, “Dimensão X” e “Dimensão Y” é possível determinar o volume de betão associado.

Para o IFC 2x3 e *Autodesk Revit*, a entidade “*IfcRelAssociatesMaterial*” apresenta um indicador único correspondente à entidade “*IfcColumnType*”. Esta entidade, assim como “*IfcColumn*”, apresenta um caminho para obter as propriedades geométricas dos elementos estruturais, o qual até é menor do que aquele conseguido através de “*IfcColumn*”. No entanto, como foi anteriormente referido, esta entidade nem sempre é gerada e, por isso, o trajeto primário a considerar será aquele alcançado através da entidade “*IfcColumn*”. O mesmo se aplica a todos os elementos que possuam estas supraentidades.

Quadro 3.4 – Trajeto para a obtenção das propriedades geométricas de Pilares.⁴

VERSÃO	ENTIDADE DO ELEMENTO ESTRUTURAL	TRAJETO	DIFERENÇAS ENTRE VERSÕES
IFC 2x3	<i>IfcColumn</i>	<i>IfcProductDefinitionShape</i>	⊕ <i>IfcMappedItem</i>
		<i>IfcShapeRepresentation</i>	⊕ <i>IfcRepresentationMap</i>
		<i>IfcMappedItem</i>	⊕ <i>IfcShapeRepresentation</i>
		<i>IfcRepresentationMap</i>	
		<i>IfcShapeRepresentation</i>	
		<i>IfcExtrudedAreaSolid</i>	
		<i>IfcRectangleProfileDef</i>	
IFC 4	<i>IfcColumn</i>	<i>IfcProductDefinitionShape</i>	⊖ <i>IfcMappedItem</i>
		<i>IfcShapeRepresentation</i>	⊖ <i>IfcRepresentationMap</i>
		<i>IfcExtrudedAreaSolid</i>	⊖ <i>IfcShapeRepresentation</i>
		<i>IfcRectangleProfileDef</i>	

⁴ Realça-se que as diferenças constatadas entre as versões IFC em análise poderão não ser totalmente confiáveis uma vez que o ficheiro em IFC 4 foi concebido manualmente e com o menor número de entidades possível e, por isso, a sua geração através de uma ferramenta computacional poderá levar a trajetos diferentes.

```

RELAÇÕES PARA OBTER AS PROPRIEDADES GEOMÉTRICAS: VIGAS (IFC 2x3 exemplo)
#1207= IFCRELAASSOCIATESMATERIAL ('2GzO2fuLP75BB_CXDb1KMY', #41, $, $, (#740, #841, #907,
                                     #973), #743);
#740= IFCBEAM ('18KVJfJGL26B_mTRg$Y$1V', #41, 'M_Concrete-Rectangular Beam
             existente:25 x 40:159641', $, 'M_Concrete-Rectangular Beam
             existente:25 x 40:159718', #711, #738, '159641');
#738= IFCPRODUCTDEFINITIONSHAPE ($, $, (#735, #729) );
#735= IFCSHAPEREPRESENTATION (#70, 'Axis', 'Curve2D', (#733));
#733= IFCPOLYLINE ((#9, #731));
#731= IFCARTESIANPOINT ((4.75, 0.));
#729= IFCSHAPEREPRESENTATION (#72, 'Body', 'SweptSolid', (#719));
#719= IFCEXTRUDEDAREASOLID (#715, #718, #19, 4.74999999987808);
#715= IFCRECTANGLEPROFILEDEF (.AREA., '25 x 40', #714, 0.4,
                               0.249999999999999);
(idem para #841, #907, #973)

RELAÇÕES PARA OBTER AS PROPRIEDADES GEOMÉTRICAS: VIGAS (IFC 4 exemplo)
#769= IFCRELAASSOCIATESMATERIAL ('21kVXT9hb7C80NqRf1ZZF4', #5, $, $, (#351, #361, #371,
                                     #381, #490, #495, #540, #548), #768);
#490= IFCBEAM ('1zt8DTaE821keQ8FPj5gye', #5, 'Beam1', $, $, #489, #486, $, .BEAM.);
#486= IFCPRODUCTDEFINITIONSHAPE ($, $, (#484));
#484= IFCSHAPEREPRESENTATION (#202, 'Body', 'SweptSolid', (#483));
#483= IFCEXTRUDEDAREASOLID (#476, #191, #13, 5.2);
#476= IFCRECTANGLEPROFILEDEF (.AREA., '20x40', #190, 0.2, 0.4);
(idem para #495, #540, #548)

```

Figura 3.24 – Relações para obter as propriedades geométricas das Vigas para os modelos: *Autodesk Revit* (IFC 2x3) e *Concebido Manualmente* (IFC 4).

As entidades que contêm as propriedades geométricas das vigas continuam a ser as mesmas dos pilares, nomeadamente “*IfcExtrudedAreaSolid*” e “*IfcRectangleProfileDef*”. Contudo, existem diferenças no trajeto percorrido entre a entidade que representa a viga (“*IfcBeam*”) e estas duas entidades comparativamente ao que sucede nos pilares para o caso prático em IFC 2x3. Para as vigas, no que diz respeito à entidade “*IfcProductDefinicionShape*”, esta contém dois indicadores únicos que remetem para duas entidades “*IfcShapeRepresentation*” distintas. Analisando estas duas entidades, observa-se que aquela que é relevante para o trajeto pretendido é a que contém no segundo atributo (“Identificador de Representação”) a nomenclatura “*Body*”. Além disso, verifica-se um trajeto mais curto para as vigas comparativamente àquele constatado para os pilares.

Um outro facto constatado a partir da Figura 3.24 é a viga ter sido modelada com uma secção de 0,25x0,40 e, no entanto, no ficheiro IFC apresentar uma secção de 0.249999999999999x0,40. Apesar de ser um erro pequeno, na ordem de 10^{-15} , entendeu-se que seria relevante mencioná-lo.

```

RELAÇÕES PARA OBTER AS PROPRIEDADES GEOMÉTRICAS: LAJE (IFC 2x3 exemplo)
#1192= IFCREASSOCIATESMATERIAL('3Vi_M8J1nCEhwBkFgjLtsU',#41,$,$, (#1031),#1084);
#1031= IFCSLAB('18KVJfJGL26B_mTRg$Y_Hx',#41,'Foundation Slab:30:159805',$,
'Foundation Slab:30',#1015,#1029,'159805',.FLOOR.);
#1029= IFCPRODUCTDEFINITIONSHAPE($,$, (#1027));
#1027= IFCSHAPEREPRESENTATION(#72,'Body','SweptSolid', (#1023));
#1023= IFCEXTRUDEDAREASOLID (#1019,#1022,#19,0.3);
#1019= IFCRECTANGLEPROFILEDEF (.AREA.,'30',#1018,4.74999999939042,
4.74999999999999);

RELAÇÕES PARA OBTER AS PROPRIEDADES GEOMÉTRICAS: LAJE (IFC 4 exemplo)
#825= IFCREASSOCIATESMATERIAL ('31kvXT9hb7C80NqRflZzF6',#5,$,$, (#390,#614),#824);
#614= IFCSLAB ('1srmDN46zBVupUF9xps7Ev',#5,'Laje1',$,$,#613,#604,$,.FLOOR.);
#604= IFCPRODUCTDEFINITIONSHAPE($,$, (#602));
#602= IFCSHAPEREPRESENTATION(#202,'Body','SweptSolid', (#601));
#601= IFCEXTRUDEDAREASOLID (#600,#191,#13,0.22);
#600= IFCRECTANGLEPROFILEDEF (.AREA.,$, $, 5.3, 5.4);
    
```

Figura 3.25 - Relações para obter as propriedades geométricas da Laje para os modelos: Autodesk Revit (IFC 2x3) e Concebido Manualmente (IFC 4).

```

RELAÇÕES PARA OBTER AS PROPRIEDADES GEOMÉTRICAS: SAPATAS (IFC 2x3 exemplo)
#1201= IFCREASSOCIATESMATERIAL('1MQvkJTDcex2B1KxZaaOT',#41,$,$, (#509,#581,#629,
#677),#512);
#509= IFCFOOTING('18KVJfJGL26B_mTRg$Y$dX',#41,'M_Footing-Rectangular:80 x 80 x 40:
159143',$,'M_Footing-Rectangular:80 x 80 x 40',#487,#507,
'159143',.NOTDEFINED.);
#507= IFCPRODUCTDEFINITIONSHAPE($,$, (#505));
#505= IFCSHAPEREPRESENTATION(#72,'Body','SweptSolid', (#495));
#495= IFCEXTRUDEDAREASOLID (#491,#494,#19,0.4);
#491= IFCRECTANGLEPROFILEDEF (.AREA.,'80 x 80 x 40',#490,0.8,0.8);
(idem para #581, #629, #677)

RELAÇÕES PARA OBTER AS PROPRIEDADES GEOMÉTRICAS: SAPATAS (IFC 4 exemplo)
#412= IFCFOOTING ('1we2ARoE50netWORKs4eva',#5,'Sapata 1',$,$,#411,#407,$,
.PAD_FOOTING.);
#407= IFCPRODUCTDEFINITIONSHAPE ($,$, (#406));
#406= IFCSHAPEREPRESENTATION (#202,'Body','SweptSolid', (#405));
#405= IFCEXTRUDEDAREASOLID (#404,#191,#13,0.75);
#404= IFCRECTANGLEPROFILEDEF (.AREA.,'150x200',#190,1.5,2.0);
(idem para #416, #420, #424)
    
```

Figura 3.26 - Relações para obter as propriedades geométricas das Sapatas para os modelos: Autodesk Revit (IFC 2x3) e Concebido Manualmente (IFC 4).

Para a laje tem-se exatamente o mesmo trajeto entre a entidade que a representa (“*IfcSlab*”) e as entidades onde são armazenadas as suas propriedades geométricas comparativamente às duas versões IFC expostas.

Finalmente, para as sapatas tem-se relações instituídas de modo análogo às da laje, diferindo apenas na entidade representativa do elemento estrutural, “*IfcFooting*”. Além disso, no exemplo em estudo em IFC 4, as sapatas não se encontram associadas a uma entidade que as relacione com um material. No entanto, tal pode ser concretizado através da entidade “*IfcRelAssociatesMaterial*”, como apresentado anteriormente. No Quadro 3.5 são apresentados os trajetos para obtenção das propriedades geométricas das vigas, da laje e das sapatas, os quais são idênticos ao nível de entidades abrangidas.

Quadro 3.5 – Comparação entre os trajetos das vigas, lajes e sapatas e pilares para IFC 2x3.

VERSÃO	ENTIDADE DO ELEMENTO ESTRUTURAL	TRAJETO	DIFERENÇAS ENTRE TRAJETOS DAS VIGAS, LAJES E SAPATAS E PILARES
IFC 2x3	<i>IfcBeam</i>	<i>IfcProductDefinitionShape</i>	⊖ <i>IfcMappedItem</i>
IFC 4	<i>IfcSlab</i>	<i>IfcShapeRepresentation</i>	⊖ <i>IfcRepresentationMap</i>
	<i>IfcFooting</i>	<i>IfcExtrudedAreaSolid</i> <i>IfcRectangleProfileDef</i>	⊖ <i>IfcShapeRepresentation</i>
IFC 2x3	<i>IfcColumn</i>	<i>IfcProductDefinitionShape</i>	⊕ <i>IfcMappedItem</i>
		<i>IfcShapeRepresentation</i>	⊕ <i>IfcRepresentationMap</i>
		<i>IfcMappedItem</i>	⊕ <i>IfcShapeRepresentation</i>
		<i>IfcRepresentationMap</i>	
		<i>IfcShapeRepresentation</i>	
		<i>IfcExtrudedAreaSolid</i> <i>IfcRectangleProfileDef</i>	

Relativamente ao *Tekla Structures*, as entidades e os trajetos atrás apresentados não estão, na totalidade, presentes nos modelos IFC gerados a partir desta ferramenta. Uma forma de obtenção das propriedades geométricas é através das entidades “*IfcQuantityLength*”, “*IfcQuantityArea*”, “*IfcQuantityVolume*”, determinando qual o elemento estrutural que lhes está associado, uma vez que nestas entidades são indicados os valores da profundidade, área da secção e volume de um determinado elemento, respetivamente. Refere-se, ainda, que estas entidades não são encontradas nem no modelo concebido manualmente em IFC 4, nem no modelo em IFC 2x3 concebido através do *Autodesk Revit*. Além disso, nestes dois modelos IFC, não são encontradas referências aos volumes dos elementos, apesar de ser possível visualizar tal informação através do ambiente de trabalho de uma ferramenta BIM, após importação do ficheiro IFC.

```

DETERMINAÇÃO: PROFUNDIDADE, ÁREA E VOLUME - PILARES (IFC 2x3 exemplo)
#876= IFCRELDEFINESBYPROPERTIES('1vd72IDCv1gx2QDSIPkC8T',#5,$,$, (#458,#411,#363,
#308),#344);
#458= IFCCOLUMN('1JHcGg0000834pEJSn CZ8n',#5,'PILAR','250*250','250*250',#455,
#457,'ID5346642a-0000-0020-3133-393731323231');
#344= IFCELEMENTQUANTITY('1gz232n053kA2M2rSvxVPh',#5,'BaseQuantities',$,$,
(#340,#341,#342,#343));
#340= IFCQUANTITYLENGTH('Length',$,$,3000.);
#341= IFCQUANTITYAREA('OuterSurfaceArea',$,$,3.125);
#342= IFCQUANTITYVOLUME('NetVolume',$,$,0.1875);
▶(idem para #411, #363, #308)

```

Figura 3.27 - Determinação: Profundidade, Área e Volume - Pilares (Tekla Structures IFC 2x3).

```

DETERMINAÇÃO: PROFUNDIDADE, ÁREA E VOLUME - VIGAS (IFC 2x3 exemplo)
#889= IFCRELDEFINESBYPROPERTIES('3s944S0MX3mfPVZEECKVRV',#5,$,$, (#691,#642,#594,
#540),#576);
#691= IFCBEAM('1JHcGg0000C34pEJSn CZ8n',#5,'VIGA','250*400','250*400',#688,#690,
'ID5346642a-0000-0030-3133-393731323231');
#576= IFCELEMENTQUANTITY('0aUW36ViD3nvpZFmyhywF8',#5,'BaseQuantities',$,$, (#572,
#573,#574,#575));
#572= IFCQUANTITYLENGTH('Length',$,$,4750.);
#573= IFCQUANTITYAREA('OuterSurfaceArea',$,$,6.375);
#574= IFCQUANTITYVOLUME('NetVolume',$,$,0.475);
▶(idem para #642, #594, #540)

```

Figura 3.28 - Determinação: Profundidade, Área e Volume - Vigas (Tekla Structures IFC 2x3).

```

DETERMINAÇÃO: PROFUNDIDADE, ÁREA E VOLUME - LAJE (IFC 2x3 exemplo)
#902= IFCRELDEFINESBYPROPERTIES('2P0I2MmRz2LhLdTngs1KLI',#5,$,$, (#772),#808);
#772= IFCSLAB('1JHcGg0000D34pEJSn CZ8o',#5,'LAJE BETAO','200*4750','200*4750',
#736,#771,'ID5346642a-0000-0034-3133-393731323232',.FLOOR.);
#808= IFCELEMENTQUANTITY('3Zz5dq_2TAcvfs56G6Zm$L',#5,'BaseQuantities',$,$,
(#804,#805,#806,#807));
#804= IFCQUANTITYLENGTH('Width',$,$,200.);
#805= IFCQUANTITYVOLUME('NetVolume',$,$,4.5125);
#806= IFCQUANTITYAREA('NetArea',$,$,22.5625);

```

Figura 3.29 - Determinação: Profundidade, Área e Volume - Laje (Tekla Structures IFC 2x3).

Pelas figuras anteriores é possível inferir que o modo de determinação da profundidade da peça, área da secção e volume é idêntico para pilares, vigas e laje. Além da entidade que associa elementos com igual material, “*IfcRelAssociatesMaterial*”, tem-se agora, uma outra que associa elementos com iguais

propriedades geométricas, nomeadamente “*IfcRelDefinesByProperties*”. Na Figura 3.30, são apresentados os atributos que compõem as entidades onde estão armazenadas as informações pretendidas.

ENTIDADES QUE CONTÊM: PROFUNDIDADE, ÁREA E VOLUME	(IFC 2x3 genérico)
#n= IFCQUANTITYLENGTH (“Nome”, “Descrição”, “Unidade”, “Valor da Profundidade”, “Fórmula”);	
#n= IFCQUANTITYAREA (“Nome”, “Descrição”, “Unidade”, “Valor da Área”, “Fórmula”);	
#n= IFCQUANTITYVOLUME (“Nome”, “Descrição”, “Unidade”, “Valor do Volume”, “Fórmula”);	
ENTIDADES QUE CONTÊM: PROFUNDIDADE, ÁREA E VOLUME	(IFC 4 genérico)
#n= IFCQUANTITYLENGTH (“Nome”, “Descrição”, “Unidade”, “Valor da Profundidade”, “Fórmula”);	
#n= IFCQUANTITYAREA (“Nome”, “Descrição”, “Unidade”, “Valor da Área”, “Fórmula”);	
#n= IFCQUANTITYVOLUME (“Nome”, “Descrição”, “Unidade”, “Valor do Volume”, “Fórmula”);	

Figura 3.30 -Entidades que contêm os valores da Profundidade, Área e Volume.

Do IFC 2x3 para o IFC 4 é adicionado o atributo “Fórmula” no final de cada uma das entidades apresentadas na Figura 3.30. Portanto, é no último atributo destas entidades para o IFC 2x3 e no penúltimo para o IFC 4 que são apresentadas as informações referentes à profundidade, área da secção e volume.

Apesar desta forma de cálculo de quantidades ser mais simples, não pode ser aplicada às sapatas. As entidades “*IfcRelDefinesByProperties*” que as contêm, não conduzem às mesmas entidades. Aliás, estes elementos estruturais não estão sequer associados às entidades “*IfcElementQuantity*”, “*IfcQuantityLength*”, “*IfcQuantityArea*” e “*IfcQuantityVolume*”, como se observa pela figura seguinte.

DETERMINAÇÃO: PROFUNDIDADE E ÁREA - SAPATAS	(IFC 2x3 exemplo)
#871= IFCRELDEFINESBYPROPERTIES ('0DuK4Borv2wAPyl7_5Lhrj', #5, \$, \$, \$, (#227), #243);	
#227= IFCFOOTING ('1JHcGg0000434pEJSn CZ8n', #5, 'SAPATA', '800*800', '800*800', #224, #226, 'ID5346642a-0000-0010-3133-393731323231', .NOTDEFINED.);	
#243= IFCPROPERTYSET ('1INYNLZe962wgR6KJ4qtJO', #5, 'Tekla Common', 'Propriedades Comuns para elementos do edificio partilhados', (#228, #229, #230, #231, #232, #233, #234, #235, #236, #237, #238, #239, #240, #241, #242));	
#241= IFCPROPERTYSINGLEVALUE ('Profile', \$, IFCLABEL ('800*800'), \$);	
#872= IFCRELDEFINESBYPROPERTIES ('3v0jroPb6kvlwvo3HF\$RM', #5, \$, \$, \$, (#227), #257);	
#227= IFCFOOTING ('1JHcGg0000434pEJSn CZ8n', #5, 'SAPATA', '800*800', '800*800', #224, #226, 'ID5346642a-0000-0010-3133-393731323231', .NOTDEFINED.);	
#257= IFCPROPERTYSET ('1SQUHHFbfDXB9oVBRad4mj', #5, 'Tekla Quantity', 'Quantity Properties to Shared building elements', (#244, #245, #246, #247, #248, #249, #250, #251, #252, #253, #254, #255, #256));	
#256= IFCPROPERTYSINGLEVALUE ('Length', \$, IFCLENGTHMEASURE (800.), \$);	
(idem para os pares (#863, #862), (#865, #866) e (#868, #869))	

Figura 3.31 - Determinação: Profundidade e Área - Sapatas (Tekla Structures IFC 2x3).

Apesar de existirem entidades “*IfcRelDefinesByProperties*” associadas às sapatas, nenhuma destas direciona para as entidades pretendidas. Possivelmente, esta situação constitui um erro que deve ser retificado através da instituição de relações análogas àquelas apresentadas para pilares, vigas e lajes.

Após investigação de outra forma de determinação de quantidades através do ficheiro IFC em análise, constatou-se que o volume do elemento pode ser obtido através da entidade “*IfcRelAssociatesMaterial*”. Todavia, tal metodologia não é tão objetiva quanto aquelas verificadas anteriormente. Isto porque, em vez de entidades que nos indiquem a informação relativa as propriedades geométricas dos elementos, tem-se entidades que remetem para pontos cartesianos que, juntos, compõem a superfície do elemento.

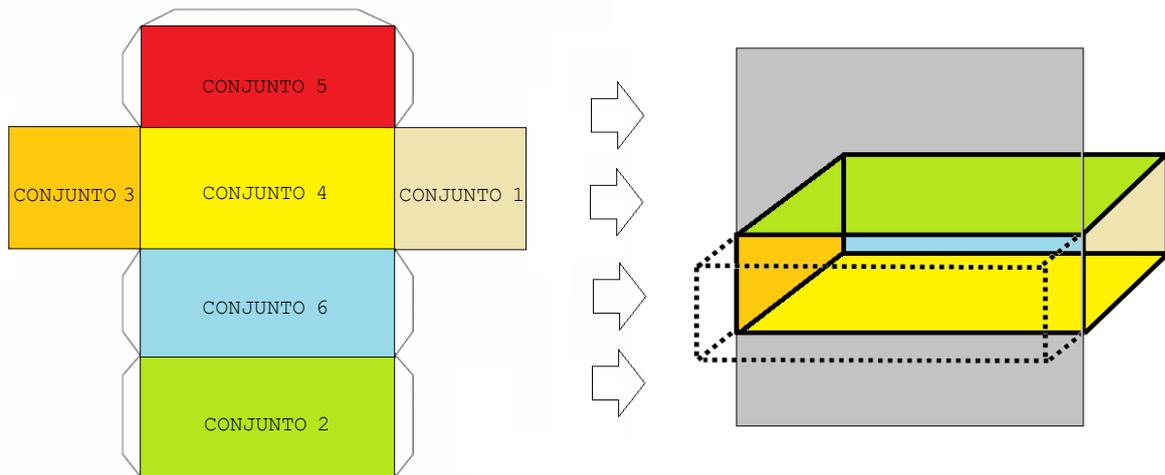
```
VOLUME DAS SAPATAS (IFC 2X3 genérico)
#906= IFCRELAASSOCIATESMATERIAL('206ElqK3bE$emE9aZ9ezWm', #5, $, $, (#227, #179, #131,
#82), #113);
#227= IFCFOOTING('1JHcGg0000434pEJSn CZ8n', #5, 'SAPATA', '800*800', '800*800', #224,
#226, 'ID5346642a-0000-0010-3133-393731323231', .NOTDEFINED.);
#226= IFCPRODUCTDEFINITIONSHAPE($, $, (#225));
#225= IFCSHAPEREPRESENTATION(#12, 'Body', 'Brep', (#78));
#78= IFCFACETEDBREP(#77);
#77= IFCCLOSEDSHELL((#57, #62, #67, #70, #73, #76));
#57= IFCFACE((#56));
#56= IFCFACEOUTERBOUND(#55, .T.);
#55= IFCPOLYLOOP((#51, #52, #53, #54));
#51= IFCCARTESIANPOINT((0., 400., -400.));
#52= IFCCARTESIANPOINT((0., 400., 400.));
#53= IFCCARTESIANPOINT((800., 400., 400.));
#54= IFCCARTESIANPOINT((800., 400., -400.));
#62= IFCFACE((#61));
#61= IFCFACEOUTERBOUND(#60, .T.);
#60= IFCPOLYLOOP((#52, #58, #59, #53));
#52= IFCCARTESIANPOINT((0., 400., 400.));
#58= IFCCARTESIANPOINT((0., -400., 400.));
#59= IFCCARTESIANPOINT((800., -400., 400.));
#53= IFCCARTESIANPOINT((800., 400., 400.));
#67= IFCFACE((#66));
#66= IFCFACEOUTERBOUND(#65, .T.);
#65= IFCPOLYLOOP((#58, #63, #64, #59));
#58= IFCCARTESIANPOINT((0., -400., 400.));
#63= IFCCARTESIANPOINT((0., -400., -400.));
#64= IFCCARTESIANPOINT((800., -400., -400.));
#59= IFCCARTESIANPOINT((800., -400., 400.));
#70= IFCFACE((#69));
#69= IFCFACEOUTERBOUND(#68, .T.);
```

```

#68= IFCPOLYLOOP ((#63,#51,#54,#64));
#63= IFCCARTESIANPOINT ((0.,-400.,-400.));
#51= IFCCARTESIANPOINT ((0.,400.,-400.));
#54= IFCCARTESIANPOINT ((800.,400.,-400.));
#64= IFCCARTESIANPOINT ((800.,-400.,-400.));
#73= IFCFACE ((#72));
#72= IFCFACEOUTERBOUND (#71,.T.);
#71= IFCPOLYLOOP ((#59,#64,#54,#53));
#59= IFCCARTESIANPOINT ((800.,-400.,400.));
#64= IFCCARTESIANPOINT ((800.,-400.,-400.));
#54= IFCCARTESIANPOINT ((800.,400.,-400.));
#53= IFCCARTESIANPOINT ((800.,400.,400.));
#76= IFCFACE ((#75));
#75= IFCFACEOUTERBOUND (#74,.T.);
#74= IFCPOLYLOOP ((#63,#58,#52,#51));
#63= IFCCARTESIANPOINT ((0.,-400.,-400.));
#58= IFCCARTESIANPOINT ((0.,-400.,400.));
#52= IFCCARTESIANPOINT ((0.,400.,400.));
#51= IFCCARTESIANPOINT ((0.,400.,-400.));
    
```

Figura 3.32 - Volume das Sapatas (Tekla Structures IFC 2x3).

Portanto, pela figura anterior pode constatar-se a presença de seis entidades “*IfcPolyLoop*”, estando cada uma destas associada a quatro pontos cartesianos. Cada um destes conjuntos de pontos cartesianos originam uma face do elemento, as quais unidas dão origem à superfície do elemento.



CONJUNTO 1	CONJUNTO 2	CONJUNTO 3	CONJUNTO 4	CONJUNTO 5	CONJUNTO 6
(0.,400.,-400.)	(0.,400.,400.)	(0.,-400.,400.)	(0.,-400.,-400.)	(800.,-400.,400.)	(0.,-400.,-400.)
(0.,400.,400.)	(0.,-400.,400.)	(0.,-400.,-400.)	(0.,400.,-400.)	(800.,-400.,-400.)	(0.,-400.,400.)
(800.,400.,400.)	(800.,-400.,400.)	(800.,-400.,-400.)	(800.,400.,-400.)	(800.,400.,-400.)	(0.,400.,400.)
(800.,400.,-400.)	(800.,400.,400.)	(800.,-400.,400.)	(800.,-400.,-400.)	(800.,400.,400.)	(0.,400.,-400.)

Figura 3.33 - Construção da superfície do elemento a partir das entidades “*IfcPolyLoop*” e “*IfcCartesianPoint*”.

3.6. MODELOS ESTRUTURAL E ARQUITETÓNICO PARA IFC 2x3 (AUTODESK REVIT)

Feita uma análise comparativa entre os modelos analíticos para as duas últimas versões IFC, segue-se um estudo incidente entre o modelo analítico anteriormente apresentado (Figura 3.8) e o modelo arquitetónico para a versão IFC 2x3 elaborado a partir do *Autodesk Revit 2014* e apresentado na Figura 3.34.



Figura 3.34 - *Autodesk Revit*. Modelo Arquitetónico.

Como referido anteriormente, esta ferramenta computacional inclui as vertentes de arquitetura e de estruturas no mesmo programa e, por isso, é possível incluir qualquer elemento estrutural no modelo arquitetónico, nem que seja apenas para fins representativos, como se efetuou no exemplo apresentado. Se os elementos estruturais apresentados possuírem apenas esta finalidade, não estando conectados entre si, então não é possível a geração automática de um modelo analítico e correspondente utilização do modelo para cálculo estrutural. Portanto, mesmo com a representação de todos os elementos estruturais, um modelo arquitetónico não substitui um modelo estrutural.

O interesse em executar uma comparação entre estes dois tipos de modelos reside em verificar se as relações entre elementos estruturais, materiais e respetiva geometria são estabelecidas do mesmo modo. Após análise do ficheiro IFC, gerado para o modelo arquitetónico, constatou-se que estas relações são semelhantes àquelas apresentadas anteriormente para o modelo estrutural.

No entanto, uma diferença notória é a desproporcionalidade de tamanho entre os ficheiros IFC gerados para os dois modelos. Enquanto o ficheiro IFC do modelo estrutural apresenta apenas 48 KB (*Kilobytes*), o do modelo arquitetónico apresenta 84.669 KB, verificando-se, por isso, uma diferença com uma ordem de grandeza superior a 1/1000. Tendo em conta que a garagem concebida constitui um pequeno projeto, então, um ficheiro IFC com um modelo arquitetónico de uma obra real pode adquirir dimensões significativas. Além disso, denota-se, também, uma diferença no número de entidades geradas nos ficheiros IFC com ordem de grandeza de cerca de 1/2200. Ressalva-se, no entanto, que o modelo arquitetónico apresenta elementos de geometria não regular como a vegetação e o carro e, por isso, de geometria muito mais complexa do que a das estruturas. Acrescenta-se, também, que cada ferramenta BIM exporta para IFC de forma distinta, podendo, portanto, obter-se para o mesmo modelo diferentes tamanhos de ficheiro.

Um critério chave para a obtenção de quantidades de estruturas a partir do IFC é a utilização do modelo estrutural em prole do modelo arquitetónico, visto que, o tamanho de um ficheiro pode influenciar a viabilidade de uma ferramenta de cálculo automático de quantidades. Como o ficheiro IFC do usuário terá de ser armazenado temporariamente para posterior cálculo, então, deverá existir um limite para o seu tamanho. Além disso, quanto menor for a sua dimensão, mais eficiente será a ferramenta idealizada.

3.7. MODELOS ARQUITETÓNICOS PARA IFC 2x3

Apesar de se ter concluído que a diferença de tamanhos nos ficheiros IFC pode ser determinante para o objetivo desta dissertação, considera-se relevante executar, também, uma analogia entre os dados gerados para os modelos arquitetónicos concebidos a partir do *Autodesk Revit* e *ArchiCAD*.

Uma vez que a organização da informação (relevante para a obtenção de quantidades) dos ficheiros IFC, relativos aos modelos arquitetónico e estrutural obtidos via *Autodesk Revit* se processa de igual modo, a sua estrutura não vai ser novamente exposta aquando da comparação com o *ArchiCAD*. De igual modo, será também interessante introduzir a análise realizada para o *Tekla Structures*, caso esta apresente semelhanças.

Desta forma, importou-se o ficheiro IFC referente ao modelo arquitetónico construído de raiz através do *Autodesk Revit* para o *Graphisoft ArchiCAD*, sendo a sua representação 3D exposta na figura seguinte.

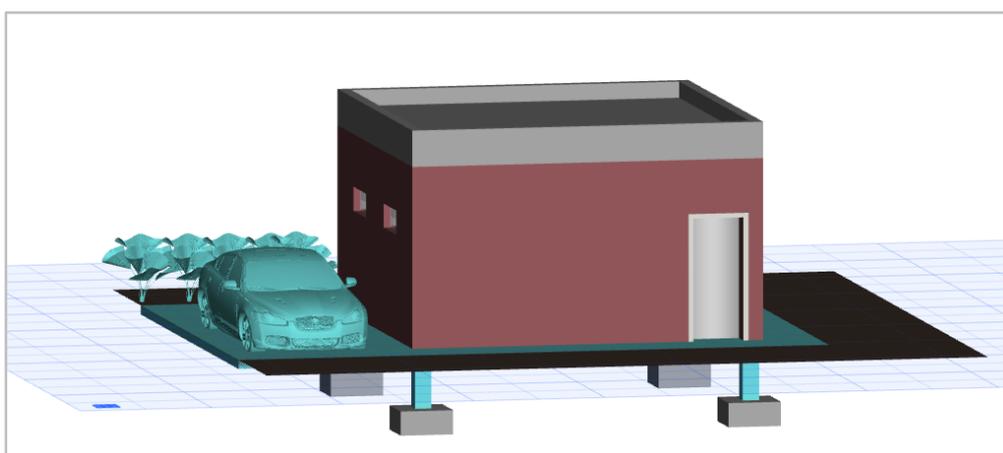


Figura 3.35 – *Graphisoft ArchiCAD*: Modelo Arquitetónico.

Numa primeira abordagem, constata-se que nem todas as famílias do modelo original foram importadas corretamente. Tal observação salienta que a transferência de dados a partir do IFC não pode ainda ser considerada como uma realidade efetiva. Posteriormente, este modelo foi convertido, também, para IFC 2x3 a partir do *ArchiCAD*.

Neste caso específico, não faz sentido executar comparações entre os tamanhos dos ficheiros IFC e entre o número de entidades dos modelos, pois a utilização do IFC pode ter conduzido à perda de informação além daquela fisicamente observável, podendo, por isso, induzir em erro. Todavia, é interessante retirar ilações do modo como a organização da informação se processa ao nível da identificação de elementos, unidades e geometria de elementos em comparação com o modelo arquitetónico do *Autodesk Revit* e o modelo estrutural do *Tekla Structures*.

3.7.1. IDENTIFICAÇÃO DE ELEMENTOS ESTRUTURAIS

O *ArchiCAD* apresenta todas as supraentidades representativas de elementos estruturais possíveis para esta versão do IFC 2x3, assim como havia apresentado o *Tekla Structures* para o modelo estrutural. O único *software* que não gerou todas estas supraentidades foi o *Autodesk Revit*. Relativamente às entidades principais representativas dos elementos estruturais não se observaram alterações.

3.7.2. UNIDADES

As unidades de comprimento constatadas no ficheiro IFC criado a partir do *ArchiCAD* são idênticas àquelas observadas para o *Tekla Structures* (Figura 3.18), ou seja, independentemente da unidade do projeto estas são convertidas sempre em milímetros, estando também associadas a uma entidade “*IfcUnitAssignment*”.

3.7.3. DIMENSÕES GEOMÉTRICAS E MATERIAIS

Assim como o *Autodesk Revit*, o *ArchiCAD* utiliza a entidade “*IfcRelAssociatesMaterial*” para relacionar elementos estruturais com o mesmo material. Relativamente ao modo de obtenção das propriedades geométricas dos elementos, observou-se uma tendência de padronização de processos semelhante àquela observada através do *Autodesk Revit*. O trajeto verificado para os pilares, vigas e sapatas através do *ArchiCAD* é idêntico ao trajeto identificado para as vigas, laje e sapatas obtido a partir do *Autodesk Revit*. Este processo é esquematizado de seguida.

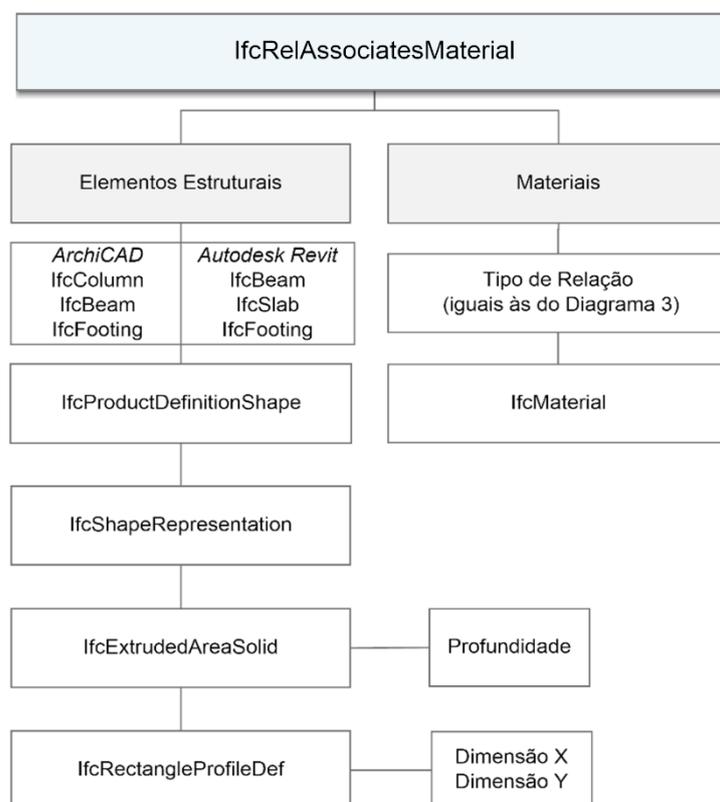


Figura 3.36 – Trajeto genérico para obtenção das propriedades dos elementos estruturais (*Autodesk Revit* vs *ArchiCAD*).

Relembra-se que para o *Autodesk Revit*, verificou-se um trajeto maior para obtenção das propriedades geométricas dos pilares, o que não ocorre para o *ArchiCAD*. No entanto, o processo verificado para obter as propriedades geométricas da laje, no *ArchiCAD*, difere do processo genérico da Figura 3.36, como se demonstra na Figura 3.37.

```

DETERMINAÇÃO: PROFUNDIDADE, ÁREA E VOLUME - VIGAS (IFC 2X3 exemplo)
#1697= IFCRELAASSOCIATESMATERIAL('3wNZe05eR87XWeEHY1aA5w', #15, $, $, (#1686), #1696);
#2578127= IFCSLAB('0qxsjxW9rBLuDUH9yEc$un', #15, 'Basic Roof:Warm Roof - Concrete:
149421', $, 'Basic Roof:Warm Roof - Concrete', #2578087, #2578124,
'34EF6B7B-809D-4B57-835E-449F0E9BFE31', .ROOF.);
#2578124= IFCPRODUCTDEFINITIONSHAPE($, $, (#2578119));
#2578119= IFCSHAPEREPRESENTATION(#67, 'Body', 'SweptSolid', (#2578109));
#2578109= IFCEXTRUDEDAREASOLID(#2578099, #2578106, #2578107, 333.);
#2578099= IFCARBITRARYCLOSEDPROFILEDEF(.AREA., '#2578097);
#2578097= IFCPOLYLINE((#2578089, #2578091, #2578093, #2578095,
#2578089));
#2578089= IFCCARTESIANPOINT((0., 0.));
#2578091= IFCCARTESIANPOINT((0., -4749.99999939));
#2578093= IFCCARTESIANPOINT((4750., -4749.99999939));
#2578095= IFCCARTESIANPOINT((4750., 0.));
    
```

Figura 3.37 - Determinação: Profundidade, Área - Laje (*ArchiCAD* IFC 2x3).

Como se pode observar pela figura anterior, em vez da entidade “*IfcExtrudedAreaProfileDef*” encaminhar para a entidade “*IfcRectangleProfileDef*” com as dimensões X e Y da laje, esta remete para a entidade “*IfcArbitraryClosedProfileDef*” que, por sua vez, faz ligação com pontos cartesianos, representados pela entidade “*IfcCartesianPoint*”. A ordem de apresentação dos indicadores únicos nesta entidade é crucial para obter a forma geométrica da laje. Neste caso em específico, tem-se o seguinte raciocínio (ao qual equivale uma superfície retangular):

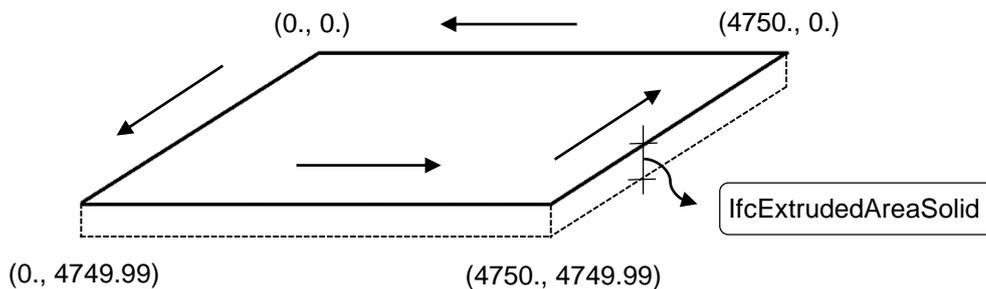


Figura 3.38 - Forma da laje (*ArchiCAD* IFC 2x3).

Para o cálculo da área da laje através das suas coordenadas cartesianas poderá ser aplicado o algoritmo de *Shoelace*. Este algoritmo permite determinar a área de qualquer polígono com as coordenadas dos seus vértices [90].

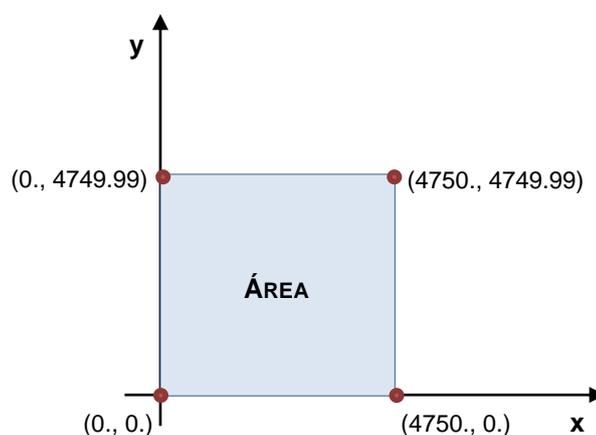
$$Area = \frac{1}{2} \left| \sum_{i=1}^{n-1} x_i \cdot y_{i+1} + x_n \cdot y_1 - \sum_{i=1}^{n-1} x_{i+1} \cdot y_i - x_1 \cdot y_n \right|$$

$$= \frac{1}{2} |x_1 \cdot y_2 + x_2 \cdot y_3 + \dots + x_{n-1} \cdot y_n + x_n \cdot y_1 - x_2 \cdot y_1 - x_3 \cdot y_2 - \dots - x_n \cdot y_{n-1} + x_1 \cdot y_n|$$

em que: “n” e “i” representam o número de lados e vértices do polígono, respetivamente.

Fórmula 3.1 – Algoritmo de *Shoelace*.

A aplicação deste algoritmo para os pontos cartesianos da laje pode ser visualizado na Figura 3.39.



$$A = \left| \frac{(0 \times 4749.99 - 0) + (0 \times 4749.99 - 4749.99 \times 4750) + (4750 \times 0 - 4749.99 \times 4750) + (4750 \times 0 - 0)}{2} \right|$$

$$= 22.562.452,5 \text{ mm}^2$$

Figura 3.39 - Cálculo da área pelo Algoritmo de *Shoelace*.

3.7.4. ELEMENTOS ADICIONAIS A ESTUDAR

Apesar de na presente dissertação apenas ser executado um levantamento de alguns dos principais elementos estruturais comuns à generalidade dos edifícios, numa futura abordagem poderão ser incluídos outros elementos relevantes para quantificação de quantidades. No Quadro 3.6 são apresentados alguns destes elementos, apresentando-se as respetivas entidades no modelo IFC.

Quadro 3.6 - Elementos adicionais passíveis de ser incluídos neste estudo [91].

ENTIDADE	ELEMENTO
<i>IfcWall</i>	Paredes
<i>IfcBuildingElementPart</i>	Ex: Isolamento de Paredes; Componentes de madeira incluídas em elementos de construção.
<i>IfcPlate</i>	Chumbadouros
<i>IfcPile</i>	Ex: Elementos de madeira, betão ou aço incorporados no solo para suportar cargas.
<i>IfcRailing</i>	Muros exteriores
<i>IfcBuildingElementProxy</i>	Elementos que o IFC ainda não consegue identificar ou elementos para os quais não existe classificação semântica adicional.
<i>IfcMember</i>	Elementos estruturais projetados para transportar cargas entre ou pontos além de apoio.

Apresentam-se ainda as respetivas representações tipo destas entidades tanto para o IFC2x3 como para o IFC 4 na figura seguinte. Sempre que um atributo for representado a negrito ou a rasurado significa que este é adicionado ou eliminado na transição do IFC 2x3 para o IFC 4, respetivamente.

```

ELEMENTOS ADICIONAIS A INVESTIGAR (IFC 2x3)
#n = IfcWall ("GUID", "Historial do usuário", "Nome", "Descrição", "Tipo do objeto",
             "Posicionamento do objeto", "Representação", "Etiqueta",
             "Tipo pré-definido");

#n = IfcBuildingElementPart ("GUID", "Historial do usuário", "Nome", "Descrição",
                             "Tipo do objeto", "Posicionamento do objeto",
                             "Representação", "Etiqueta");

#n = IfcPlate ("GUID", "Historial do usuário", "Nome", "Descrição", "Tipo do objeto",
              "Posicionamento do objeto", "Representação", "Etiqueta",
              "Tipo pré-definido");

#n = IfcPile ("GUID", "Historial do usuário", "Nome", "Descrição", "Tipo do objeto",
             "Posicionamento do objeto", "Representação", "Etiqueta",
             "Tipo pré-definido", "Tipo de construção");

#n = IfcRailing ("GUID", "Historial do usuário", "Nome", "Descrição", "Tipo do
                objeto", "Posicionamento do objeto", "Representação", "Etiqueta",
                "Tipo pré-definido");

#n = IfcBuilingElementProxy ("GUID", "Historial do usuário", "Nome", "Descrição",
                             "Tipo do objeto", "Posicionamento do objeto",
                             "Representação", "Etiqueta", "Tipo pré-definido",
                             "Tipo de Composição");

#n = IfcMember ("GUID", "Historial do usuário", "Nome", "Descrição", "Tipo do
                objeto", "Posicionamento do objeto", "Representação", "Etiqueta",
                "Tipo pré-definido");

```

Figura 3.40 - Representação no modelo IFC de elementos adicionais a investigar [91].

4

PROGRAMAÇÃO E WEB SERVICE

4.1. INTRODUÇÃO

Após a concretização do estudo relativo ao IFC segue-se a construção de um modelo, associado a linguagem programática, que permita a obtenção das propriedades de cada elemento estrutural, assim como as quantidades de materiais por elemento e por tipo de elemento. O formato do IFC para o qual esta ferramenta estará disponível é o IFC 2x3, pelos motivos já apresentados.

À quantificação de quantidades através de processos manuais associam-se tempos elevados de trabalho, assim como custos de mão-de-obra, o que não se verifica para a ferramenta projetada. Além disso, todos os erros provenientes da má contabilização de quantidades através de processos manuais podem ser evitados através desta ferramenta.

É evidente que uma grande parte das ferramentas BIM atuais apresenta já uma interface que permite o cálculo automático de quantidades. Todavia, estas ferramentas não são gratuitas, nem executam o cálculo de quantidades via IFC. A construção deste modelo permitirá a obtenção de tais valores à distância de um clique e de forma gratuita e sem exigir ao utilizador a instalação de novas aplicações.

Finalmente, projeta-se a transposição desta ferramenta para um “*Web-Service*” de modo a que qualquer profissional possa utilizá-la.

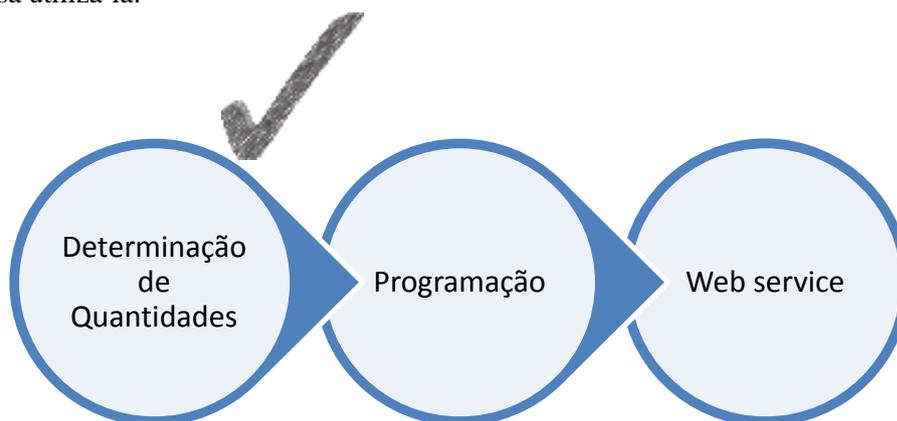


Figura 4.1 – Programação, Determinação de Quantidades e *Web service*.

4.2. PROGRAMAÇÃO

Através do estudo realizado anteriormente é agora possível iniciar a construção um modelo programático que execute o cálculo automático de quantidades. Neste sentido, foram analisadas as ferramentas programáticas existentes, tendo-se dado prioridade a todas aquelas que apresentem API's (*Application Programming Interface*) para IFC.

As API's ou, em português, Interfaces de Programação de Aplicativos são ferramentas regidas por um conjunto de regras que permitem ao utilizador compreender como utilizar uma determinada ferramenta para comunicação entre programas, aplicações ou sistemas operativos [92]. A utilização das API's possibilita a partilha de serviços, sem a necessidade de partilha total do código original, mas apenas da informação estritamente necessária para o seu acesso. Por este motivo, todo o processo torna-se mais eficiente, uma vez que não é necessário despende tempo em compreender e procurar no código original a informação pretendida. Além de que dificilmente os autores dos códigos gostariam de tornar pública a totalidade dos seus trabalhos. De acordo com Josh Walker, analista na Forrest Research Inc., a importância das API's na atualidade é tal que equipara a construção de uma aplicação sem Interfaces de Programação de Aplicativos com a construção de um edifício sem portas. A API é um meio para abrir portas e facilitar as trocas de informação [93] [94].

Além disso, as API's funcionam em segundo plano e não estão, por isso, à vista dos seus utilizadores, tornando por isso todo o processo mais simples e elegante. Por exemplo, alguns serviços da “*Google Maps*” podem ser acedidos através de uma API disponibilizada pela *Google*, o que permite, por exemplo, calcular a distância entre dois locais, utilizando apenas aplicações informáticas disponíveis nos servidores desta empresa.

Para este caso em particular foi efetuado um levantamento de API's que permitam a organização das bases de dados de texto IFC, de modo a obterem-se bibliotecas onde seja possível a leitura e manipulação da informação orientada por objetos. Os critérios chave considerados para a sua escolha são os seguintes:

- Custo;
- Linguagem programática suportada;
- Formato Aberto (i.e. com o código original incluído);
- Funções desempenhadas;
- Formatos IFC suportados;
- Última atualização.

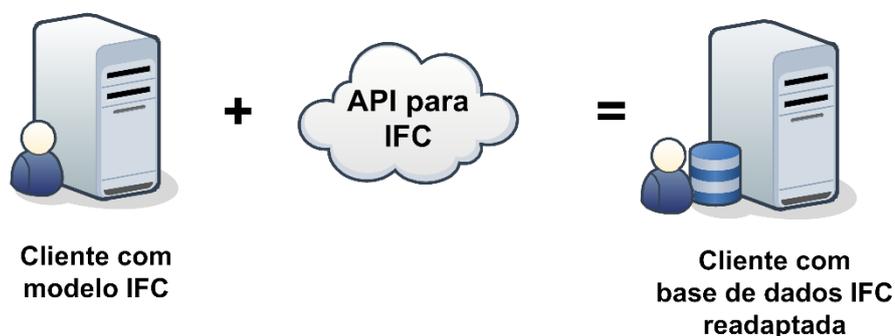


Figura 4.2 - Comunicação através de uma API para IFC.

No Anexo 1 apresentam-se todas as API's encontradas para IFC com a função desejada, assim como uma breve descrição das suas características. Após a sua análise, concluiu-se que a melhor escolha seria a utilização da API para o aplicativo *IfcOpenShell*, o qual é gratuito, em formato aberto e apesar de ser escrito em C++, existe uma transposição para a linguagem *Python*, disponível em: [95]. Contudo, para usufruir desta transposição é necessária uma compilação dos ficheiros fornecidos, utilizando para isso o programa *Visual Studio* e outros subprogramas indicados no endereço disponibilizado.

Esta é sem dúvida uma vantagem para indivíduos que tiveram pouco contanto com linguagens programáticas, pois o *Python* é uma linguagem com sintaxe simples, elegante e que permite, de forma direta, o armazenamento de qualquer conteúdo através de bases de dados. Segundo M. Costa Pinto, “Base de dados é um conjunto de dados relacionados entre si, armazenados de uma forma conjunta em estruturas tipo ficheiro, com redundância controlada, que suportam acessos diretos através de processos bem definidos ou através de programas desenvolvidos para o efeito” [96].

Por si só, todo o ficheiro IFC constitui uma base de dados com inter-relacionamento de informação através de uma estrutura padrão. Já foi demonstrado anteriormente que, através de processos manuais, é possível a sua utilização para obtenção de informações distintas. Contudo, para a sua manipulação através de processos automatizados, é necessária a reorganização/ adaptação da sua estrutura base, de acordo com metodologias que proporcionem uma fácil acessibilidade às informações através de ferramentas programáticas. As bases de dados apropriadas depois de obtidas têm em vista a obtenção de quantidades, no entanto, poderão ser facilmente empregadas para qualquer fim pretendido.

Acrescenta-se ainda que o *Python* é uma linguagem interpretada, o que pode ser encarado como uma vantagem e uma desvantagem. Vantagem porque não necessita de compilação constante e desvantagem talvez por não ser tão eficiente quanto outras linguagens existentes [97]. Contudo, as suas vantagens têm-se demonstrado mais significativas, pois a sua popularidade tem aumentado circunstancialmente. A este propósito, o “*YouTube*” é concebido nesta linguagem e os problemas que poderão advir da eficácia desta linguagem conseguiram ser ultrapassados.

Na presente dissertação, considerou-se como fator mais relevante a utilização do *Python* como ferramenta programática, devido aos motivos já referidos. Acrescenta-se que não foi utilizada a API mencionada, tendo sido, por isso, necessária a adaptação da estrutura de dados tipo do IFC. Todos os objetivos referentes a esta adaptação do IFC e conseqüente uso foram alcançados.

Antes de mais é necessário introduzir alguns conceitos sobre a estrutura utilizada para a programação em *Python*, nomeadamente funções, classes e métodos. Uma função permite reunir um conjunto de operações e executá-las todas de uma só vez, através da sua invocação. Um método é uma função, a diferença é que encontra-se dentro de numa classe. Finalmente, uma classe permite a criação de objetos definidos pelo seu utilizador. A classe pode conter informações distintas, as quais podem ser herdadas pelos métodos que a compõem. Para o programa concebido utilizou-se uma classe denominada “IFC”, a qual é composta por um conjunto de métodos que serão apresentados convenientemente.

De seguida apresenta-se o modo como o modelo concebido opera através de um fluxograma simplificado. Refere-se que todo o programa é concebido utilizando a língua inglesa de modo a abranger um público maior.

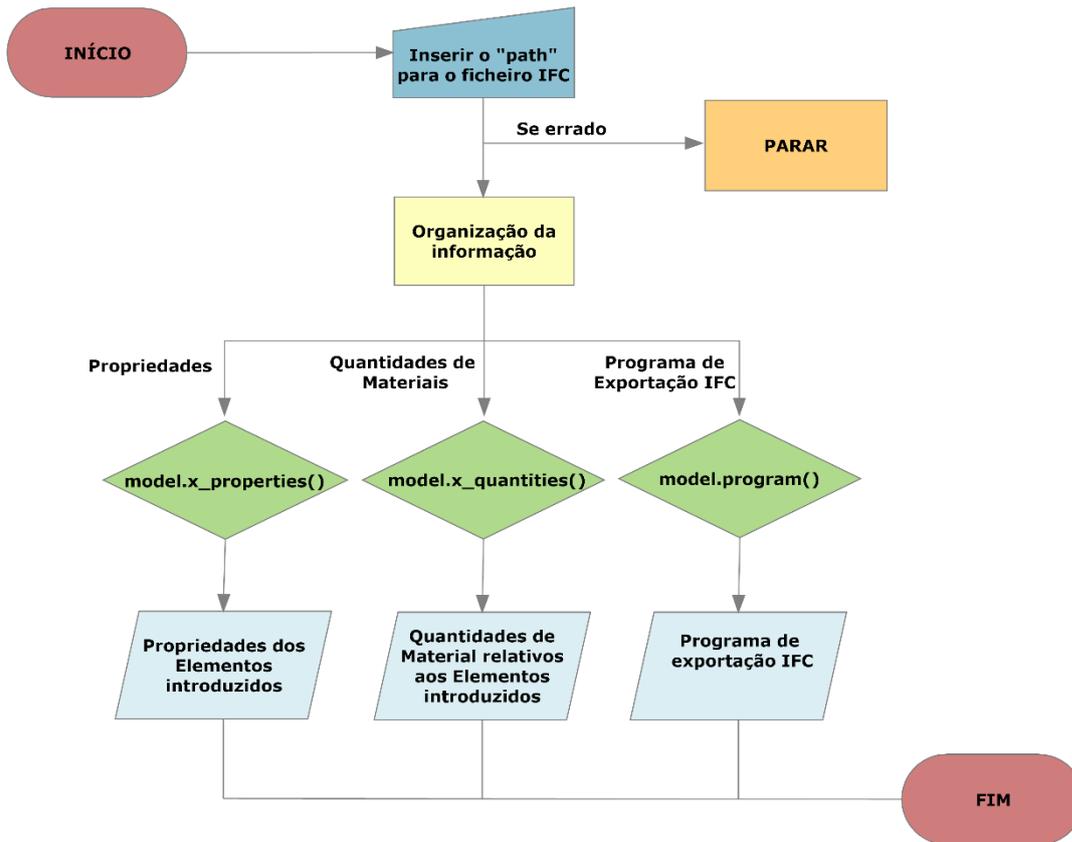


Figura 4.3 - Fluxograma Simplificado.

Portanto, a organização da informação para posterior manipulação é uma propriedade da própria classe. Todas as restantes operações/ cálculos são realizados recorrendo a métodos distintos.

No Anexo 2 pode ser consultado um fluxograma mais pormenorizado do programa desenvolvido, o qual inclui nos processos de decisão quais os comandos a introduzir e, posteriormente, o respetivo resultado. As várias fases/ etapas que compõem este programa são apresentadas de seguida, juntamente com exemplos práticos, de modo a obter-se uma melhor perceção do seu funcionamento.

1º Após correr o programa surge uma mensagem para introduzir o “path” para o ficheiro IFC:

```

    INSERIR O "PATH" DO FICHEIRO
    What is the path to your file?
    C:\Users\hp-pc\Desktop\Dropbox\MIEC14JPBIM\modeloRevit.ifc (exemplo)
  
```

Figura 4.4 - Inserir o path do ficheiro IFC.

Apenas após introduzir um “path” referente a um ficheiro de texto é que é possível a passagem à etapa seguinte. Caso o “path” não faça referência a um ficheiro deste tipo, surgirá a mensagem “Error, can't find file or read data. Please try again.” e, por isso, será necessário iniciar o programa novamente.

2º Feito isto, o ficheiro IFC é convertido em duas bases de dados distintas, cada com uma estrutura de um dicionário.

Em *Python*, o conteúdo de um dicionário é representado entre chavetas e apresenta a seguinte estrutura:

```
ESTRUTURA TIPO DE UM DICIONÁRIO EM PYTHON
nome dicionário = {chave 1: valor(es), chave 2: valor(es), chave n: valor(es)}
```

Figura 4.5 - Estrutura tipo de um dicionário em *Python*.

Com esta forma de organização da informação foi possível a construção dos seguintes dicionários:

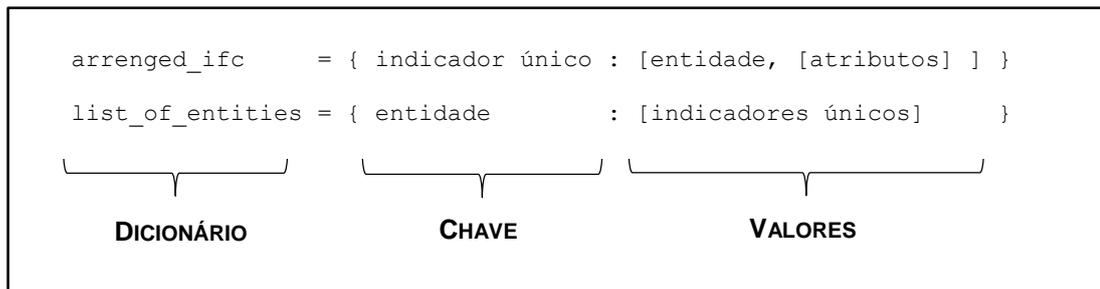


Figura 4.6 - Bases de dados criadas a partir dos ficheiros IFC.

Como se pode inferir pela figura anterior, os valores de ambos os dicionários foram agrupados através de listas (as quais são representadas entre parêntesis retos).

No dicionário “*aranged_ifc*” cada indicador único insere-se numa chave e a respetiva entidade e atributos correspondem aos valores dessa chave. Os valores são representados através de uma lista global, tendo os atributos sido incluídos numa sublista como forma de separação da entidade. Para este dicionário, será ainda utilizada uma segunda sublista, dentro da lista dos atributos, mediante a verificação da situação já constatada na Figura 3.3 e que se expõe novamente de seguida.

```
NO FICHEIRO IFC:
#203= IFCCURVESTYLEFONT('Concrete:11', (#200,#201,#202));

NO DICIONÁRIO "ARANGED_IFC":
{'#203' : ['IFCCURVESTYLEFONT', ['Concrete:11', [#200,#201,#202] ] ]}
```

Figura 4.7 - Exemplo genérico da conversão da informação IFC para o dicionário “*aranged_IFC*”.

Logo, a utilização de sublistas na lista dos atributos será apenas necessária para atributos com múltiplas informações.

Relativamente ao dicionário “*list_of_entities*” foi concebido com o objetivo de reunir todos os indicadores únicos que partilham a mesma entidade. Este dicionário, não só permite contabilizar o número de repetições de cada entidade (possibilitando, por isso, verificar se existe o mesmo número de elementos estruturais no projeto e no ficheiro IFC), mas também permite a sua utilização como um complemento ao dicionário “*aranged_IFC*” para a manipulação da informação.

3º Após introdução do “path”, o ficheiro IFC é manipulado de modo a obterem-se as informações relevantes para este estudo. O acesso ao seu resultado final pode ser conseguido através da introdução dos comandos apresentados no quadro seguinte.

Quadro 4.1 - Comandos para obtenção dos resultados finais (Python).

COMANDO A INTRODUIZIR PARA OBTER:				
ELEMENTO	PROPRIEDADES	QUANTIDADES	PROGRAMA DE EXPORTAÇÃO	EXEMPLO
Programa	-	-	model.program()	Figura 4.8
Pilares	model.columns_properties()	model.columns_quantities()	-	Figura 4.9 Figura 4.10
Vigas	model.beams_properties()	model.beams_quantities()	-	-
Lajes	model.slabs_properties()	model.slabs_quantities()	-	-
Sapatas	model.footings_properties()	model.footings_quantities()	-	-
Elementos Estruturais	model.elements_properties()	model.elements_quantities()	-	-
Materiais	-	model.material_quantities()	-	Figura 4.11

Portanto, os comandos podem ser divididos em três grupos distintos, nomeadamente propriedades, quantidades e programa de exportação. De seguida, expõem-se os exemplos referenciados na tabela anterior utilizando o ficheiro IFC referente ao modelo estrutural exportado a partir do *Autodesk Revit*.

COMANDO: "model.program()"	
Resultado:	{'Program': 'Autodesk Revit 2014 (ENU)'} }

Figura 4.8 - Chave: "model.program()".

Portanto, pela Figura 4.8, ao inserir-se a chave “model.program()” obtém-se um dicionário onde o valor do dicionário é representado entre plicas e corresponde ao programa onde foi gerado o ficheiro IFC em análise. Refere-se que qualquer texto em *Python* (string) é reproduzido desta forma (i.e. entre plicas).

As propriedades de cada elemento podem ser obtidas introduzindo os comandos: “model.columns_properties” para os pilares, “model.beams_properties” para as vigas, “model.slabs_properties” para as lajes, “model.footings_properties” para as sapatas e finalmente “model.elements_properties” para o conjunto de todos os elementos anteriormente referidos. Na figura seguinte, apresenta-se o resultado obtido após introdução do comando relativo às propriedades dos pilares.

```

CÓDIGO: "model.columns_properties( )"

Resultado:
{'Columns': [['IFCCOLUMN', 'Concrete-Square-Column(1):25 x 25:158880', 'Concrete - Cast-in-Place Concrete', 'Meters', 0.25, 0.25, 3.0, 'm2', 0.1, 'm3', 0.2], ['IFCCOLUMN', 'Concrete-Square-Column(1):25 x 25:158882', 'Concrete - Cast-in-Place Concrete', 'Meters', 0.25, 0.25, 3.0, 'm2', 0.1, 'm3', 0.2], ['IFCCOLUMN', 'Concrete-Square-Column(1):25 x 25:158884', 'Concrete - Cast-in-Place Concrete', 'Meters', 0.25, 0.25, 3.0, 'm2', 0.1, 'm3', 0.2], ['IFCCOLUMN', 'Concrete-Square-Column(1):25 x 25:158886', 'Concrete - Cast-in-Place Concrete', 'Meters', 0.25, 0.25, 3.0, 'm2', 0.1, 'm3', 0.2]]}

Comentário:
Pelo resultado anterior observa-se a existência de quatro pilares cada um com as seguintes características:

[Entidade, Nome, Material, Unidade de Comprimento, Dimensão X, Dimensão Y, Profundidade, Unidade de Área, Área, Unidade de Volume, Volume]

Exemplo:
['IFCCOLUMN', 'Concrete-Square-Column(1):25 x 25:158880', 'Concrete - Cast-in-Place Concrete', 'Meters', 0.25, 0.25, 3.0, 'm2', 0.1, 'm3', 0.2]

```

Figura 4.9 - Chave: "model.columns_properties()".

Relativamente às quantidades de materiais, estas podem ser obtidas através dos comandos: "model.columns_quantities()" para os pilares, "model.beams_quantities()" para as vigas, "model.slabs_quantities()" para as lajes, "model.footings_quantities()" para as sapatas, "model.elements_quantities()" para o conjunto dos elementos estruturais referidos (com divisão por elemento) e "model.material_quantities()" para quantidades por material sem distinção por elementos. Apresenta-se na Figura 4.10 o resultado obtido para as quantidades dos pilares.

```

CÓDIGO: "model.columns_quantities( )"

Resultado:
{'Concrete - Cast-in-Place Concrete': ['m3', 0.8]}

Comentário:
A estrutura padrão para os comandos referentes a quantidades por tipo de elemento é a seguinte:

{Material : [Unidade de Volume, Volume]}

Caso o comando inserido fosse "model.elements_quantities( )", ou seja, para todos os elementos encontrados com distinção por tipo de elemento, ter-se-ia uma estrutura do tipo:

{Tipo de elemento: [[Material, [Unidade de Volume, Volume]]]}

Por exemplo:
{'Beams': [['Concrete - Precast Concrete', ['m3', 2.0]], 'Slabs': [['Concrete - Cast In Situ - Bet\\X2\\00E3\\X0\\o', ['m3', 6.8]], 'Columns': [['Concrete - Cast-in-Place Concrete', ['m3', 0.8]], 'Footings': [['Concrete - Cast In Situ - Bet\\X2\\00E3\\X0\\o', ['m3', 1.2]]]}

```

Figura 4.10 - Código: "model.columns_quantities()".

Finalmente, tem-se o comando referente às quantidades por material sem distinção por tipo de elemento (“*model.material_quantities()*”), sendo o seu resultado apresentado na Figura 4.11.

```

CÓDIGO: "model.material_quantities( )"

Resultado:
{'Concrete - Precast Concrete': ['m3', 2.0], 'Concrete - Cast In Situ -
Bet\\\\X2\\\\00E3\\\\X0\\\\o': ['m3', 6.8], 'Concrete - Cast In Situ -
Bet\\\\X2\\\\00E3\\\\X0\\\\o': ['m3', 1.2], 'Concrete - Cast-in-Place Concrete': ['m3',
0.8]}

Comentário:
A estrutura deste dicionário é idêntica à estrutura apresentada para as quantidades
por tipo de elemento singularmente, isto é:
{Material : [Unidade de Volume, Volume]}
    
```

Figura 4.11 - Código: "*model.material_quantities ()*".

Como se pode observar, todos os resultados obtidos são representados sob a forma de um dicionário. Esta decisão foi tomada por uma questão de facilidade de execução do próximo objetivo, respeitante à implementação do *Web service*.

Todos os resultados apresentados nos dicionários, referentes a dimensões lineares, superfícies e volumes foram arredondados de acordo com o Quadro 4.2, o qual foi extraído do livro “Curso sobre Regras de Medição na Construção” já mencionado.

Quadro 4.2 - Unidades e Arredondamentos a adotar [52].

UNIDADES BASE DE MEDIDA		
UNIDADE	DESIGNAÇÃO	SÍMBOLO
Genérica	Unidade	un
Comprimento	Metro	m
Superfície	Metro quadrado	m ²
Volume	Metro cúbico	m ³
ARREDONDAMENTOS DE RESULTADOS PARCIAIS		
MEDIDA	ARREDONDAMENTO	
Metro (m)	Centímetro (cm)	
Metro quadrado (m ²)	Decímetro quadrado (dm ²)	
Metro cúbico (m ³)	Decímetro cúbico (dm ³)	
ARREDONDAMENTOS DE RESULTADOS GLOBAIS		
MEDIDA	ARREDONDAMENTO	
Metro (m)	Decímetro (dm)	
Metro quadrado (m ²)	Decímetro quadrado (dm ²)	
Metro cúbico (m ³)	Decímetro cúbico (dm ³)	

Apenas o resultado obtido através de “*model.material_quantities()*” é alvo de arredondamentos para resultados globais. Todos os resultados relativos aos restantes comandos (à exceção de “*model.program()*”) são arredondados considerando-os como resultados parciais.

Há ainda que salientar que a utilização deste programa é apenas viável para modelos IFC referentes a modelos estruturais de uma obra. Testes com o modelo arquitetónico estudado revelaram que a quantidade de informação que este contém, condiciona tempo de cálculo dos resultados pretendidos e o próprio funcionamento do programa. Enquanto para o modelo estrutural o cálculo é automático, para o modelo arquitetónico, devido à quantidade de informação existente, duas horas não foram suficientes para a obtenção de um resultado.

4.3. WEB SERVICE

No subcapítulo anterior foi realizado um estudo de API's existentes para a manipulação do IFC, sendo os serviços prestados por estas, obtidos através da instalação de *softwares* que correm de forma interna no computador. O presente *Web service* apresenta, também, uma API como guia de utilização de serviços, sendo o acesso ao *Web service* conseguido a partir do seguinte endereço: “*openg.fe.up.pt*”.

Um *Web service* não é um *Website*, pois ao contrário de um *Website*, este é direcionado a programas e não a seres humanos. Apesar disto, estes conceitos estão diretamente relacionados, uma vez que é através de um *Web service* que é possível a comunicação entre um programa e um *Website*. Os programas utilizados podem ser construídos de raiz ou, se existirem *Web services* similares disponíveis na internet, estes poderão ser diretamente invocados, através das respetivas API's, para o *Website*. A utilização de um *Web service* implica a disponibilização de um serviço remoto, acedido via internet através de um navegador, onde é fornecido um mecanismo de pedido/ resposta.

Por exemplo, a conceção de um *Website* que após introdução de um nome de uma cidade retorne a temperatura dessa mesma cidade, pode ser conseguido através de um *Web service* já existente que estabeleça ligação com uma estação meteorológica. Neste sentido, também o *Web service* idealizado poderá ser invocado por outras aplicações, sejam elas *Websites* ou *Web services*.

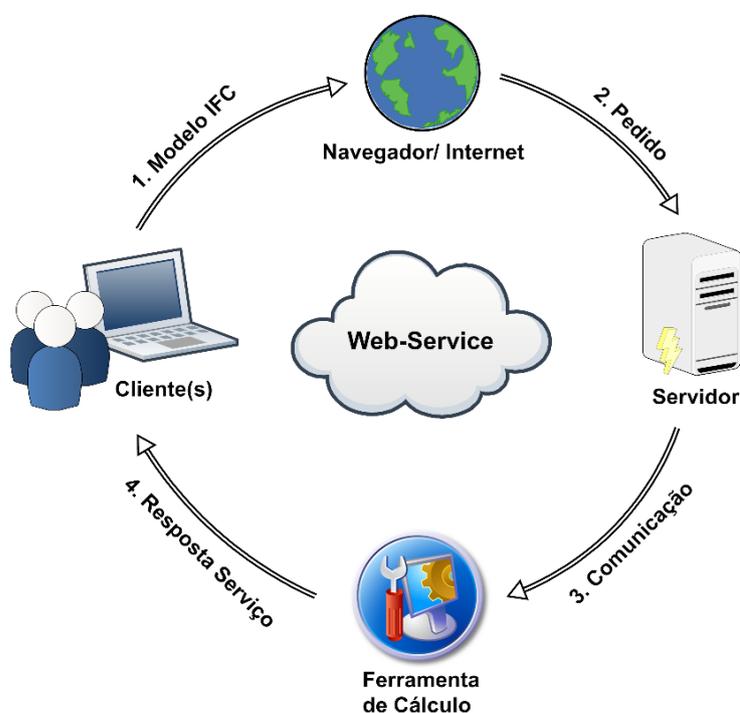


Figura 4.12 - Funcionamento do *Web service* para IFC.

Para a elaboração de um *Web service* que possibilite a utilização do programa concebido para IFC é necessária a existência de um servidor remoto que receba os pedidos dos clientes, estabeleça comunicação com a ferramenta de cálculo e retorne respostas ao serviço solicitado, como exemplificado na figura anterior.

A ferramenta de cálculo desenvolvida em *Python*, apresentada no subcapítulo 4.1. utiliza o “*path*” de um ficheiro IFC para contabilizar quantidades. Por este motivo, existe uma limitação no que diz respeito à localização do ficheiro utilizado, uma vez que este tem, obrigatoriamente, de se encontrar num ambiente local. Neste sentido, é necessária uma readaptação do programa, de modo a que o parâmetro de passagem do pedido do cliente seja um endereço referente a um ficheiro alocado na internet e não um “*path*” de um ficheiro armazenado localmente. Este endereço tem que encaminhar para uma página onde o modelo IFC seja diretamente descarregado e não para uma em que seja necessário clicar numa opção de “*download*” (via indireta). Por exemplo, a “*Dropbox*” utiliza esta via indireta para partilha de ficheiros.

Mais ainda, como se apresentou anteriormente, conceberam-se diferentes métodos, em *Python*, para a determinação de informações específicas no ficheiro IFC. Por uma questão de simplicidade de processos, todos os resultados relevantes (dicionários) serão incorporados num só dicionário, de modo a constituir-se um “*output*” global. Os dicionários a serem incluídos neste dicionário global são aqueles obtidos através dos comandos apresentados na figura seguinte.

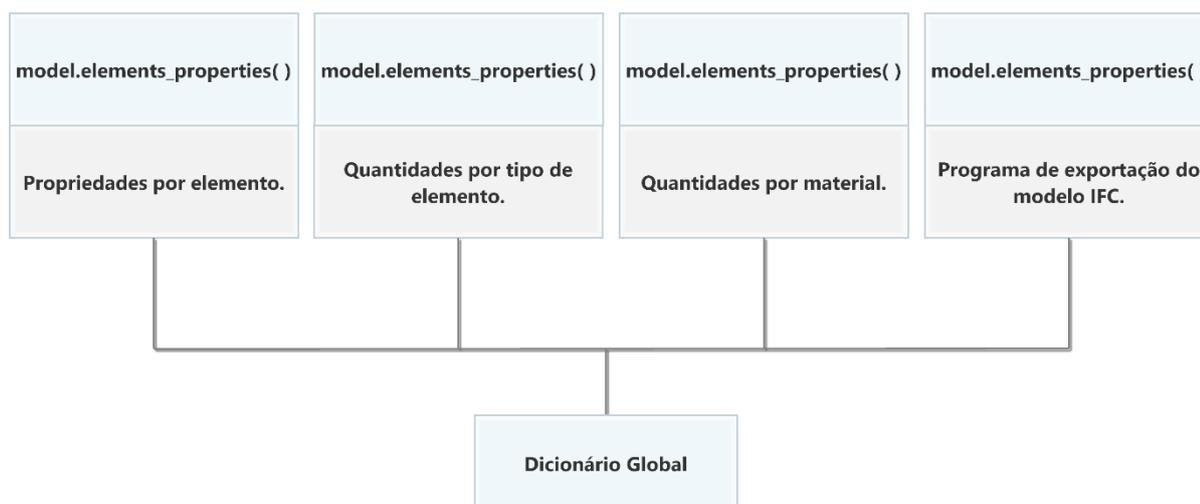


Figura 4.13 – Dicionários que compõem o dicionário global.

Por sua vez, a informações contidas no dicionário global foram convertidas para estruturas XML, através do *Python*, para posterior manipulação. Enquanto o HTML possibilita a conceção de *Websites* através da representação da informação pretendida, o XML possibilita o seu transporte e armazenamento. Por este motivo, pode-se proferir que o XML funciona como um complemento ao HTML.

No presente estudo, recorre-se ao *Microsoft Excel 2013* para a representação do XML. Nesta versão deste programa existem duas vias para atingir tal objetivo, nomeadamente através do conjunto de funções “*WEBSERVICE*” e “*FILTERXML*” (Figura 4.15, 4.16 e 4.17) ou através da importação direta do ficheiro XML com a opção “*From XML Data Import*” (Figura 4.18 e 4.19). Refere-se que enquanto a primeira permite determinar informações particulares, a segunda importa o XML como um todo. Acrescenta-se que a estrutura XML concebida para cada uma das duas vias apresentadas

é distinta. Com efeito, são adicionados dois métodos distintos ao programa em *Python* que permitem a conversão do dicionário global para as respetivas estruturas XML, nomeadamente:

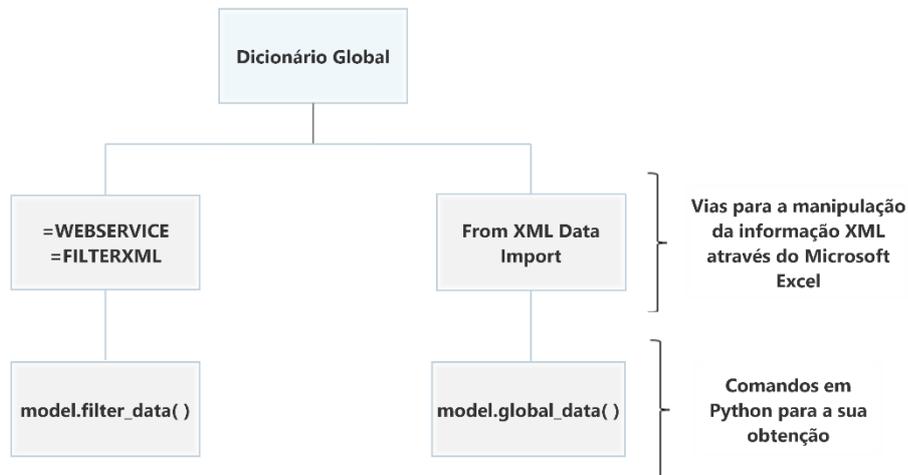


Figura 4.14 – Comandos em *Python* para a conversão da informação do dicionário global para XML.

Apresentam-se de seguida indicações para a utilização destas duas formas de exploração das capacidades do *Microsoft Excel 2013*. Para a via da esquerda da figura anterior, tem-se:

```

FUNÇÕES "WEBSERVICE" e "FILTERXML": MICROSOFT EXCEL 2013
=WEBSERVICE("endereço do Web Service com o respetivo pedido")
=FILTERXML(xml; xpath)
  
```

Figura 4.15 – Funções "WEBSERVICE" e "FILTERXML" do *Microsoft Excel 2013*.

A função "WEBSERVICE" permite invocar o *Web Service* pretendido sem a necessidade de aceder a um navegador padrão. Para a sua utilização, e para o presente *Web Service*, basta indicar um endereço com a seguinte tipologia:

```

Exemplo de utilização da função "WEBSERVICE"
=WEBSERVICE(openg.fe.up.pt/api/ifc/filtered?address=http://goo.gl/USwWxP)

O atributo da função "WEBSERVICE" é composto pelas seguintes componentes:
1) Endereço do Web Service: "openg.fe.up.pt";
2) Acesso ao serviço com o tipo de tratamento de dados pretendido, nomeadamente:
   - "/api/ifc/filtered" para filtração de dados;
   - "/api/ifc/global" para importação global da informação.
3) Representação do endereço: "?adress=";
4) Endereço do ficheiro IFC alocado remotamente: "http://goo.gl/USwWxP".

RESULTADO OBTIDO:
Obtém-se o conjunto de caracteres que compõem a estrutura XML.
  
```

Figura 4.16 - Exemplo de utilização da função "WEBSERVICE".

Relativamente à função “FILTERXML”, os seus parâmetros de passagem são “xml” que equivale ao resultado da aplicação da função “WEBSERVICE” e “path” o qual corresponde ao caminho para obter uma informação específica no código XML. O “path” tem que, obrigatoriamente, respeitar a hierarquia estrutural do código XML, a qual é apresentada no Anexo 3 (a mesma apresenta-se em inglês pois é um guia de manipulação do XML). A função “FILTERXML” está associada a uma manipulação célula a célula, a qual pode ser exemplificada através da figura seguinte, referente a um exemplo de um modelo IFC exportado a partir do *Autodesk Revit*:

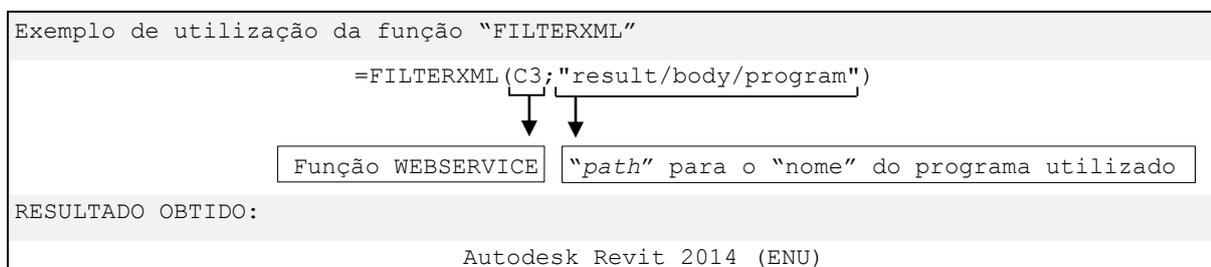


Figura 4.17 - Exemplo da aplicação da função “FILTERXML” do *Microsoft Excel 2013*.

Uma versão completa da aplicação destas funções a todo o código XML, com comentários explicativos, é apresentada no Anexo 4 e pode ser descarregada temporariamente para utilização pessoal a partir de:

- https://www.dropbox.com/s/rvu9567w3cpmz0q/calculo_medicoes.xlsx

Para a importação da totalidade do ficheiro XML, para o *Microsoft Excel 2013*, sem necessidade de manipulação célula a célula, podem seguir-se os seguintes passos:

- 1) Como referido, na Figura 4.16, substitui-se a palavra “filtered” por “global” no endereço do *Web service* com o respetivo pedido;
- 2) Adiciona-se ao endereço, no início, “view-source:” para visualizar o código XML;
- 3) Cola-se o endereço obtido na barra de endereços de um navegador padrão (Figura 4.18);
- 4) Utiliza-se, por exemplo, o programa “bloco de notas”, colando o resultado gerado num novo documento e gravando-o com a terminologia *.xml.
- 5) Posteriormente seguem-se os passos expostos na Figura 4.19 para importação do ficheiro *.xml para o *Microsoft Excel 2013*.



Figura 4.18 - Obtenção do Código XML para importação global da informação no *Microsoft Excel 2013*.

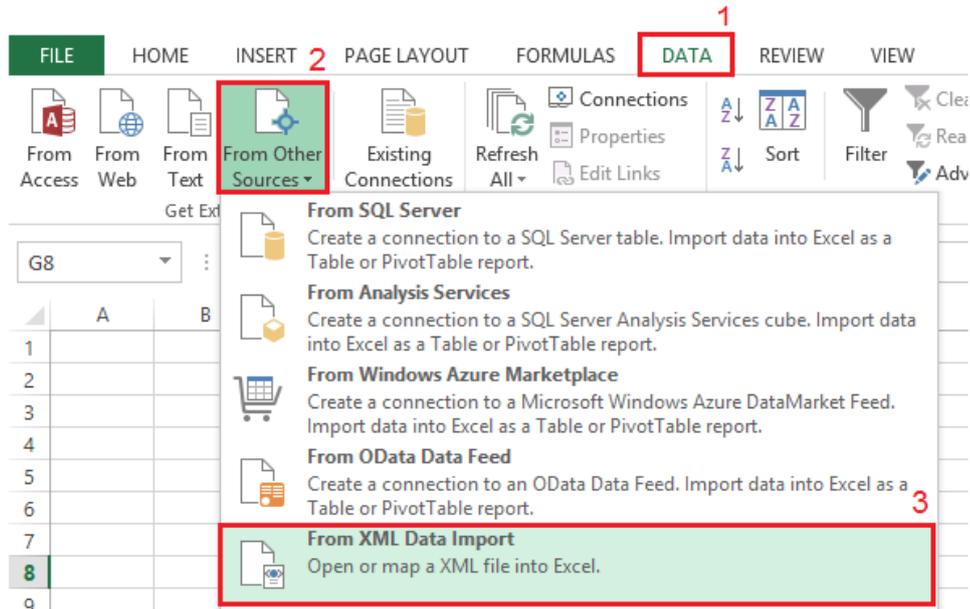


Figura 4.19 - Opção "From XML Data Import" do Microsoft Excel.

A estrutura do XML concebida para esta via tem como objetivo a conceção de uma tabela genérica, com opção de filtragem de conteúdos por coluna. No Anexo 5 é apresentada uma aplicação desta funcionalidade para o modelo IFC exportado através do *Tekla Structures*.

Todas as interdependências entre métodos, relativos à programação em *Python*, são apresentadas no Anexo 6. Neste anexo, introduzem-se já os métodos apresentados para utilização no *Web service*. Todo o código concebido em *Python* que possibilita estas operações é disponibilizado no Anexo 7.

5

CONCLUSÕES E SUGESTÕES

5.1. CONCLUSÕES

Os objetivos traçados relativos à transposição de rotinas de extração automática de quantidades de trabalhos de estruturas de betão armado, a partir de modelos IFC (versão IFC 2x3), para um modelo programático foram alcançados, recorrendo para tal à ferramenta programática *Python*. Contudo, o programa concebido abrange apenas os casos de estudo analisados na presente dissertação, conseguidos utilizando as seguintes ferramentas BIM: *Autodesk Revit*, *Tekla Structures* e *Graphisoft ArchiCAD* e respeitantes apenas aos elementos estruturais: vigas, pilares, lajes e fundações diretas.

As versões IFC analisadas são IFC 2x3 e IFC 4, ou seja, a penúltima e última versão existentes, respetivamente. A maioria das ferramentas BIM existentes no mercado ainda opera segundo as diretrizes do IFC 2x3, sendo o âmbito de aplicação do IFC 4 ainda limitado. Daí o programa desenvolvido ser respeitante ao IFC 2x3.

Os casos de estudo analisados permitiram retirar ilações sobre a estruturação-tipo de ficheiros IFC respeitantes a modelos analíticos/ estruturais e modelos arquitetónicos. Estes casos de estudo possibilitaram a comparação de ficheiros IFC relativos ao mesmo modelo virtual (estrutural ou arquitetónico) e obtidos através de diferentes ferramentas BIM. Além disso, permitiram, também, uma análise entre modelos IFC referentes a modelos estruturais e arquitetónicos. Todas as análises apresentadas são realizadas segundo uma perspetiva de obtenção de quantidades. A única comparação onde foi empregue a versão IFC 4 foi entre modelos estruturais, uma vez, que foi disponibilizado um ficheiro construído manualmente pelo Eng. Sérgio Pinho.

Das comparações realizadas a partir de diferentes ferramentas BIM, conclui-se:

a. IFC (geral)

- O IFC apresenta uma estrutura padrão com características próprias;
- Os objetos são representados nos modelos IFC através de entidades, cada uma associada a um indicador único e compostas por diversos atributos, os quais apresentam informações particulares;
- É crucial a utilização de “famílias” na elaboração de modelos virtuais, de modo a toda a informação ser organizada segundo as respetivas entidades nos ficheiros IFC.

b. Casos de Estudo IFC

- Existe uma tendência clara de evolução entre as versões IFC analisadas, no sentido de uma melhor divisão de conteúdos;
- Cada ferramenta BIM analisada apresenta formas distintas de organização da informação para o IFC;
- A cada elemento estrutural corresponde uma entidade principal, podendo existir ou não supra entidades destas entidades principais. Estas supra entidades representam elementos da mesma família (i.e. pilares, vigas, entre outros). Enquanto as entidades principais se repetem para todos os casos de estudo, as supra entidades têm uma presença inconstante de programa para programa, devendo, por isso, ser utilizadas as entidades principais para identificação de elementos;
- Os elementos estruturais e as suas propriedades são representados, no IFC, através de entidades distintas, existindo trajetos compostos por um conjunto de indicadores únicos que os interligam. Tal é possível, uma vez que no IFC os objetos herdam propriedades entre si;
- O IFC agrupa elementos com características semelhantes através de uma mesma entidade (por exemplo, elementos com o mesmo material ou unidades de projeto);
- O modo de associação de um elemento aos seus materiais pode ser alcançado através de diferentes trajetos. Existem recomendações por parte da *buildingSMART* para o modo como estes trajetos devem ser definidos (para o IFC 4), no entanto, não são de carácter obrigatório. Estes trajetos são diferentes para o IFC 2x3 e IFC 4;
- Os elementos estruturais encontram-se, também, associados às suas propriedades geométricas através de diferentes trajetos. Este estudo apenas abordou em detalhe elementos de secções retangulares, sendo o modo de obtenção das suas propriedades dependente do programa em análise. Observou-se, também, que os trajetos podem variar de elemento para elemento, com algumas diferenças pontuais entre si. Contudo, estes tendem para o mesmo trajeto genérico. As entidades onde as propriedades geométricas são armazenadas dependem da forma dos elementos e podem, também, variar de programa para programa, uma vez que são obtidas através de vias distintas: (1) de forma direta; (2) através de um conjunto de pontos cartesianos que no seu conjunto formam a área da secção do elemento;
- As unidades de projeto podem ser definidas durante todo o decorrer do projeto, mas a sua alteração tem que ser realizada através da ferramenta BIM, uma vez que não existe uma opção direta que permita escolher as unidades no momento de exportação do modelo para IFC;
- Não podem existir diferentes unidades do mesmo tipo de unidade de medição (i.e. diferentes unidades de comprimento, superfície, volume, entre outras). Por exemplo, os elementos podem ser modelados na unidade de comprimento centímetros, mas não podem existir alguns elementos em centímetros e outros em milímetros. Além disso, cada tipo de unidade é independente de outra existente e, por isso, é possível existirem simultaneamente informações no modelo IFC em m, cm² e mm³.
- Apenas o *Autodesk Revit* mostrou capacidades na utilização das unidades do modelo BIM no ficheiro IFC. Em todos os outros programas analisados as unidades são convertidas para uma mesma unidade, a qual é independentemente da unidade de projeto. Por exemplo, a unidade de comprimento do ficheiro IFC para o *Tekla Structures* e *Graphisoft ArchiCAD* é sempre o milímetro. No entanto, detetou-se uma falha na utilização da unidade decímetros a partir do *Autodesk Revit*, uma vez que no ficheiro IFC obtiveram-se alguns elementos em

que as propriedades geométricas foram alteradas para valores incorretos e não correspondentes a qualquer tipo de unidade.

- Notou-se uma desproporcionalidade de tamanho entre os ficheiros IFC gerados comparativamente a modelos estruturais e arquitetónicos, verificando-se uma diferença de tamanho e de número de entidades geradas com ordem de grandeza superior a 1/1000 e 1/2200, respetivamente. Com efeito, um modelo arquitetónico de uma obra real pode adquirir dimensões significativas. Ressalva-se, no entanto, que este pode apresentar elementos de geometria não regular como elementos paisagísticos (vegetação) e, por isso, de geometria muito mais complexa do que a das estruturas;
- Como cada ferramenta BIM exporta para IFC de forma distinta, podem obter-se para o mesmo modelo diferentes tamanhos de ficheiro;
- Um critério chave para a obtenção de quantidades de estruturas a partir do IFC é a utilização do modelo estrutural em prole do modelo arquitetónico, visto que, o tamanho de um ficheiro pode influenciar a eficiência de uma ferramenta de cálculo automático;
- Os problemas de interoperabilidade entre ferramentas BIM via IFC ainda são uma realidade, apesar de cada vez menores. A existência de diferentes formas de estruturar o mesmo modelo BIM através do IFC está na origem destes problemas.

c. Medições de betão armado através de modelos IFC

- A imposição de regras de medição padrão ao modelo IFC através de uma ferramenta de cálculo automático é inviável. As regras que deverão ser adotadas para as medições deverão ser respeitadas na fase de modelação e não numa posterior;
- As medições apenas terão em conta o modelo IFC gerado, excluindo qualquer informação adicional que possa existir, devendo, portanto, a informação ser o mais detalhada possível;
- Apesar de ser possível a constituição de um modelo 3D com a localização de cada elemento, uma vez que cada ferramenta BIM executa esta função na importação dos modelos IFC, os algoritmos matemáticos associados a esta utilidade apresentam um nível de complexidade elevado. Assim sendo, a implementação de uma ferramenta que permita o reconhecimento espacial de elementos requer algum esforço, uma vez que é necessária uma readaptação da informação do IFC, considerando todas as interdependências entre elementos. A sua utilização mostrar-se-ia importante para a quantificação de quantidades de cofragens. Sem esta ferramenta apenas é possível admitir que as cofragens correspondem a uma determinada percentagem dos volumes contabilizados;
- Apenas dando o uso apropriado à tecnologia disponível, é possível evitar a problemática da quantificação de trabalhos com critérios díspares.

d. Programação via *Python* para extração automática de quantidades de modelos IFC

- O *Python* é uma linguagem que permite, de forma direta, o armazenamento de qualquer conteúdo através de bases de dados, adotando sintaxes simples e elegantes;
- Foi demonstrado que através dos trajetos estudados via IFC é possível a obtenção de quantidades relativas a volumes. Contudo, para a manipulação da informação através de ferramentas programáticas, é necessária a reorganização/ adaptação da estrutura base do IFC, de modo a constituírem-se metodologias que proporcionem uma fácil manipulação da informação. Esta base de dados pode ser empregue para outro fim além do presente (obtenção de quantidades);

- Existem diversas aplicações que permitem a manipulação do IFC para diferentes ferramentas programáticas;
- O programa desenvolvido possibilita a obtenção de diferentes parâmetros: (1) programa de exportação do ficheiro IFC; (2) propriedades de pilares, vigas, lajes e sapatas isoladas, nomeadamente, entidade IFC, nome do elemento, material, unidade de comprimento, dimensão X, dimensão Y, profundidade, unidade de área/ superfície, área, unidade de volume e volume; (3) quantidades de materiais por tipo de elemento estrutural, incluindo o material, a unidade de volume e o volume associados; (4) quantidades de material sem diferenciação de tipo de elemento, com nome do material, unidade de volume e volume correspondente. Os arredondamentos adotados estão de acordo com a publicação “Curso sobre Regras de Medição na Construção” [52].

e. *Web service*

- A incorporação de ferramentas de cálculo num *Web service* suprime a necessidade de instalação de um programa internamente num computador, uma vez que permite aceder a serviços remotos de uma determinada ferramenta, recorrendo para tal a um navegador;
- Para tornar o *Web service* o mais simples possível todos os dados de passagem do programa foram reunidos numa única fonte de informação, constituída por toda a informação que o programa desenvolvido disponibiliza. Por sua vez, esta fonte de informação foi programada de acordo com duas estruturas XML distintas para manipulação via *Microsoft Office Excel*.

5.2. SUGESTÕES

As medições na construção associam-se a uma infinidade de casos particulares que devem ser contabilizados nas fichas de quantidades. Para o estudo desta temática via IFC são necessários estudos detalhados através de casos de estudo, constituídos a partir das mais diversas ferramentas BIM. Tal deve-se à possibilidade do IFC poder ser estruturado de formas distintas para modelos BIM semelhantes.

Relativamente às quantidades de volumes dos elementos estruturais, sugere-se o estudo das entidades expostas no Anexo 8, representativas das formas geométricas que os elementos poderão assumir, assim como das entidades referentes a outros elementos suscetíveis de serem incluídos num próximo estudo, os quais são apresentados no subcapítulo 3.7.4.

Quanto às cofragens, sugere-se a programação de uma ferramenta que permita o reconhecimento espacial dos elementos e constituição de metodologias que permitam identificação de interseções de elementos. Uma vez que as ferramentas BIM conseguem converter ficheiros IFC para modelos tridimensionais (3D), a constituição desta ferramenta é uma possibilidade.

Não menos importante deve ser, também, a execução de uma investigação respeitante à possibilidade de obtenção de quantidades de armaduras a partir de modelos IFC, a qual não foi abordada na presente dissertação.

BIBLIOGRAFIA

- [1] Montemor, M., Costa, A., Neves, J., Colaço, R., Cruz, H., Sequeira, L., Gil, L., Amaral, P., Fernandes, J., Rodrigues, A., Fredericci, C., Gonçalves, M., Diogo, A., Fernandes, J., Simões, A., Margarido, F. *Materiais de Construção Guia de Utilização*. Loja da Imagem - Marketing, Comunicação e Gestão, Lda, 2005.
- [2] Mueller, T. “*Brunelleschi’s Dome*”. National Geographic Magazine, Fevereiro de 2014.
- [3] Cruz, L. B. “*História da Engenharia Civil*”. Unidade curricular: História da Engenharia Civil, Faculdade de Engenharia da Universidade do Porto, 2008.
- [4] Eastman, C. M., Teicholz, P., Sacks, P., Liston, K. “*BIM Handbook*”. John Wiley & Sons, Lda, New Jersey, 2011.
- [5] Sousa, H., Martins, J. P. “*Gestão de Projetos: Processo Construtivo (Construction Life Cycle)*”. Unidade Curricular: Gestão de Projetos, 2011.
- [6] Kymmell, W. *Building Information Modeling: Planning and Managing Construction Projects with 4D CAD and Simulations*. The McGraw-Hill Companies, Inc, Estados Unidos da América, 2008.
- [7] Martins, J. P. “*Modelação do Fluxo de Informação no Processo de Construção*”. Dissertação de Doutoramento, Faculdade de Engenharia da Universidade do Porto, 2009.
- [8] Bernstein, P. “*Integrated Practice: It’s Not Just About the Technology*”. 2005. http://info.aia.org/SiteObjects/files/IPD_Guide_2007.pdf. 17-Jun-2014.
- [9] Love, P. E., Irani, Z. “*Evaluation of IT costs in construction*”. Automation In Construction, vol. 10, no. 6, pp. 649–658, 2001.
- [10] Langenwalter, G. A. “*Enterprise Resources Planning And Beyond: Integrating Your Entire Organization*”. CRC Press LLC, Florida, 2000.
- [11] Oliveira, M., Soeiro, A. “*Conflitos em Empreitadas de Construção – Causas, Consequências e Soluções*”. *Forum Int. Gestão da Construção – GESCON*, 11 e 12 de Dezembro de 2008, Faculdade de Engenharia da Universidade do Porto.
- [12] Meireles, A. R. “*Say ‘NO!’ to Hollywood BIM: An Advanced Integration of the BIM Process at a Builder*”. 2013. http://www.aecbytes.com/buildingthefuture/2013/BIM_MotaEngil.html. 12-Jun-2014.

- [13] <http://www.igsmelbourne.com.au/tech-explained/what-is-bim>. 06-Jun-2014.
- [14] http://en.wikipedia.org/wiki/Noble_gas#mediaviewer/File:Xenon-tetrafluoride-3D-vdW.png. 02-Abr-2014.
- [15] Jernigan, F. "*BIG BIM little bim*", 4Site Press, Salisbury, 2008.
- [16] "*Building Design: White Paper*", 2012.
- [17] Nederveen, G. A., Tolman, F. P. "*Modelling multiple views on buildings*". *Automation in Construction*, vol. 1, pp. 215–224, 1992.
- [18] Ijeh, I. "*BIM builds up: Five recent innovations in BIM*". 2011.
<http://www.building.co.uk/news/morrell-five-year-plan-to-roll-out-bim/5018275.article>. 10-Abr-2014
- [19] Engelbart, D. C. "*Augmenting Human Intellect: A Conceptual Framework*" 1962.
- [20] Bergin, M. S. "*History of BIM*" 2012.
<http://www.architectureresearchlab.com/ar1/2011/08/21/bim-history/>. 15-Jun-2014.
- [21] Eastman, C. M., Henrion, M. "*GLIDE: A Language for Design Information Systems*", Carnegie, 1977.
- [22] Green, C. "*Design Systems and Computer Programs*". 2007.
<http://greenart.info/green/cwbgcv01.htm#8>. 16-Jun-2014
- [23] Eastman, C. M. "*Building Product Models: Computer Environments Supporting Design and Construction*". Florida: CRC Press LLC, 1999.
- [24] Migilinskas, D., Ustinovichius, L. "*Computer-Aided Modelling, Evaluation and Management of Construction Projects According to PLM Concept*", *Cooperative Design, Visualization and Engineering*, pp. 242–250, 2006.
- [25] Manning, R., Messner J. I. "*Case Studies in BIM Implementation For Programming of Healthcare Facilities*", 2008.
- [26] <http://www.princeton.edu/facilities/info/news/archive/?id=8217>. 2012. 5-Jun-2014.
- [27] "*The Cost Savings of BIM*". 2009. http://therevitkid.blogspot.pt/2009_11_01_archive.html. 5-Jun-2014
- [28] Azhar, S. "*Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry*". *Leadership and Management in Engineering*, 2011.
- [29] Smith, P. "*BIM & the 5D Project Cost Manager*". *Procedia - Soc. Behav. Sci.*, vol. 119, pp. 475–484, 2014.
- [30] Tang, P., Huber, D., Akinci, B., Lipman, R., Lytle, A. "*Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques*". *Automation in Construction*, vol. 19, no. 7, pp. 829–843, 2010.

- [31] Motamedi, A., Soltani, M. M., Hammad, A. “*Localization of RFID-equipped assets during the operation phase of facilities*”, *Adv. Eng. Informatics*, vol. 27, no. 4, pp. 566–579, 2013.
- [32] Thompson, D. B. “*e-Construction: Don’t get soaked by the next wave*”. *Constr. Law Brief. Pap.*, 2001.
- [33] Rosenberg, T. L. “*Building information modeling*” 2007.
<http://wenku.baidu.com/view/d38a8834a32d7375a41780a8>. 17-Jun-2014
- [34] Miner, R., Thomson, D. “*Building Information Modeling- BIM: Contractual Risks are Changing with Technology*”. 2006. <http://www.greenrealestatelaw.com/2007/02/building-information-modeling-bim-contractual-risks-are-changing-with-technology/>. 10-Jun-2014.
- [35] Andia, A. “*Topological Future: Generative BIM Thinking*”
- [36] McGraw Hill Construction. “*The Business Value of BIM for Construction in Major Global Markets: How Contractors Around the World Are Driving Innovation With Building Information Modeling*” 2014.
- [37] WSPGroup. “*BIM around the world*”. 2013.<http://www.wspgroup.com/en/wsp-group-bim/BIM-around-the-world/>. 15-Jun-2014.
- [38] Taborda, P., Cachadinha, N. “*BIM nas obras públicas em Portugal: Condicionantes para uma implementação com sucesso*”. Congresso Construção 2012 - 4º Congresso Nacional, 18,19 e 20 de Dezembro de 2012, Coimbra, ITeCons.
- [39] Wong, K. D., Wong, K. W., Nadeem, A. “*Government roles in implementing building information modelling systems: Comparison between Hong Kong and the United States*” *Construction Innovation: Information, Process, Management*, vol. 11, pp. 61–76, 2011.
- [40] Couto, J. P. “*A Deficiente Qualidade e Segurança na Construção Continua a Ser Determinante para a sua Falta de Competitividade: Estudo em Curso sobre os Factores de Competitividade do Sector da Construção*”, Guimarães, 2006.
- [41] Succar, B. “*Building information modelling framework: A research and delivery foundation for industry stakeholders*”. *Automation In Construction*, vol. 18, pp. 357–375, 2009.
- [42] AIA National: AIA California Council. “*Integrated Project Delivery: A Guide*”. 2007.
http://info.aia.org/SiteObjects/files/IPD_Guide_2007.pdf. 17-Jun-2014.
- [43] Ernststrom, B., Hanson, D., Hill, D., Jarboe, J., Kenig, M., Nies, D., Russell, D., III, L. S., Webster, T. “*The Contractors’ Guide to BIM*” 2007.
http://www.engr.psu.edu/ae/thesis/portfolios/2008/tjs288/Research/AGC_GuideToBIM.pdf. 21-Jun-2014.
- [44] <http://buildipedia.com/aec-pros/engineering-news/baku-hoks-flame-towers>. 17-Jun-2014.
- [45] <http://www.anteprojectos.com.pt/2012/04/05/terminal-ferroviario-de-ligacao-oeste-kowloon-hong-kong/>. 17-Jun-2014.
- [46] <http://www.ice.org.uk/topics/BIM/Case-studies/Victoria-station-upgrade>. 17-Jun-2014.

- [47] <http://www.ice.org.uk/topics/BIM/Case-studies/BIM-and-the-21st-Century-Panama-Canal>. 17-Jun-2014.
- [48] Faria, J. A. "*Gestão de Obras e Segurança*". Faculdade de Engenharia da Universidade do Porto (FEUP), 2013.
- [49] Nassar, K. "*The Effect of Building Information Modeling on the Accuracy of Estimates*". American University in Cairo.
- [50] "*Decreto-Lei n.º 18/2008 de 29 de Janeiro*". Ministério das Obras Públicas, Transportes e Comunicações. Imprensa Nacional-Casa da Moeda - Lisboa, 2008.
- [51] "*Portaria n.º 959/2009 de 21 de Agosto*". Ministério das Obras Públicas, Transportes e Comunicações. Imprensa Nacional-Casa da Moeda - Lisboa, 2009.
- [52] "*Curso Sobre Regras de Medição na Construção*". 5ª ed. Laboratório Nacional de Engenharia Civil (LNEC), Lisboa, 2000.
- [53] "*Pronic Sistema de Geração e Gestão de Informação Técnica para Cadernos De Encargos*". 2008. <http://www2.inescporto.pt/cese/noticias-eventos/nos-na-imprensa/pronic-sistema-de-geracao-e-gestao-de-informacaotecnica-para-cadernos-de-encargos/>. 22-Jun-2014.
- [54] "*ProNIC*". GEQUALTEC, 2011. http://paginas.fe.up.pt/~gequaltec/w/index.php?title=ProNIC#Antecedentes_Pr.C3.B3ximos. 22-Jun-2014.
- [55] Sousa, H., Mêda, P., Carvalho, P. "*Electronic procurement on construction works — offer evaluation methodologies*". 9th European Conference on Product and Process Modelling, 2012.
- [56] Monteiro, A., Martins, J. P. "*A survey on modeling guidelines for quantity takeoff-oriented BIM-based design*". Automation in Construction, vol. 35, pp. 238–253, 2013.
- [57] Lee, S. K., Kim, K. R., Yu, J. H. "*BIM and ontology-based approach for building cost estimation*". Automation in Construction, vol. 41, pp. 96–105, 2014.
- [58] Zhiliang, M., Zhenhua, W., Wu, S., Zhe, L. "*Application and extension of the IFC standard in construction cost estimating for tendering in China*". Automation in Construction, vol. 20, no. 2, pp. 196–204, 2011.
- [59] BuildingSMART. "*IFC Overview summary*". <http://www.buildingsmart-tech.org/specifications/ifc-overview>. 08-Maio-2014.
- [60] Thein, V. "*Industry Foundation Classes (IFC) - BIM Interoperability Through a Vendor-Independent File Format*". 2011. http://ftp2.bentley.com/dist/collateral/docs/bentley_institute/White_paper_IFC.pdf. 12-Jun-2014.
- [61] Granholm, L. "*BIM, IT in construction, history, present and future*" 2011.

- [62] “ISO 10303-21 (*Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange*)”. 2013.
- [63] SCRA-STEP. "STEP Application Handbook - ISO 10303". p. 175, 2006.
- [64] Bazjanac, V. “*The Implementation of Industry Foundation Classes in Simulation Tools for the Building Industry*”.
http://www.inive.org/members_area/medias/pdf/Inive/IBPSA/UFSC585.pdf. 22-Abr-2014
- [65] Liebich, T. “*IFC4 the new buildingSMART Standard*”. 2013. http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc4-release/buildingSMART_IFC4_WhatisNew.pdf. 21-Abr-2014
- [66] GEQUALTEC. “*Industry Foundation Classes*” 2013.
http://paginas.fe.up.pt/~gequaltec/w/index.php?title=IFC#cite_note-IAI-1. 19-Abr-2014.
- [67] IAI. “*IFC Object Model Architecture Guide*”. *International Alliance of Interoperability*, 1999.
http://iaiweb.lbl.gov/Resources/IFC_Releases/IFC_Release_2.0/FINAL_Documents/IFC_R2_ObjectModelArchitectureGuide_US.PDF. 19-Abr-2014.
- [68] Pinho, S. “*O Modelo IFC como Agente de Interoperabilidade: Aplicação ao domínio das estruturas*”. Dissertação de Mestrado. Faculdade de Engenharia da Universidade do Porto (FEUP), 2013.
- [69] Sousa, H., Martins, J. P. “*Projecto SIGABIM*”. Faculdade de Engenharia da Universidade do Porto, 2011.
- [70] IAI. “*BuildingSMART International Limited - Industry Foundation Classes*” 2010.
<http://www.iai-tech.org/>. 19-Abr-2014.
- [71] Cavanaugh, E. “*WhitePaper - Web services: Benefits, challenges, and a unique, visual development solution*”. 2006. <http://www.altova.com/whitepapers/webservices.pdf>. 20-Maio-2014.
- [72] W3C. “*HTTP - Hypertext Transfer Protocol*”. <http://www.w3.org/Protocols/>. 15-Jun-2014.
- [73] Pautasso, C., Zimmermann, O., Leymann, F. “*RESTful Web Services vs. ‘Big’ Web Services: Making the Right Architectural Decision*”. 2008.
- [74] Macherla, P. “*Types of Web services – Big and RESTful*”. 2012.
<http://theopentutorials.com/tutorials/web-services/types-of-web-services-big-and-restful/>. 15-Jun-2014.
- [75] Berners-Lee, H. F., Fielding, R. “*Hypertext Transfer Protocol -- HTTP/1.0*”. 1996.
http://www.hjp.at/doc/rfc/rfc1945.html#sec_1. 15-Jun-2014.
- [76] Berners-Lee, T., Fielding, R., Masinter, L. “*Uniform Resource Identifier (URI): Generic Syntax*” 2005. <http://tools.ietf.org/html/rfc3986>. 15-Jun-2014.

- [77] Freed, N., Borenstein, N. “*Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*”. 1996. <http://www.hjp.at/doc/rfc/rfc2045.html>. 16-Jun-2014.
- [78] Fielding, R. T. “*Architectural Styles and the Design of Network-based Software Architectures*” University of California, 2000.
- [79] “*How to build a RESTful Web API on a Raspberry PI in Javascript*”. 2013. <http://thefloppydisk.wordpress.com/2013/05/08/how-to-build-a-restful-web-api-on-a-raspberry-pi-in-javascript/>. 30-Abr-2014
- [80] Barros, R., Granado, J., Rio, J., Castro, J. M. “*Technologies, Development Of Structural Applications Using Web Based*”. Faculdade de Engenharia da Universidade do Porto (FEUP), 2013.
- [81] Beckett, H. “*The business benefits of Web services*” *ComputerWeekly*, 2002. <http://www.computerweekly.com/feature/The-business-benefits-of-Web-services>.
- [82] Macherla, P. “*Introduction to Web services*” 2012. <http://theopentutorials.com/tutorials/web-services/introduction-to-web-services/>. 15-Jun-2014.
- [83] LNEC. “*Regras de Medição*”. Nelson de Vasconcelos Montes, Lisboa, 1970.
- [84] Nour, M. “*A STEP ISO-10303 Parser*”. *16th Forum Bauinformatik*, 2004.
- [85] Nagy, L., Ouellette, J., Vincent, M., Smart, J., Vazquez, A., Van Cuong, H., Thorsby, A., Torres, J., Bridwell, A., Mañas, X. L., Apted, K., Munoz, A., Lock, J., McFarlane, B., Rees, R., Carter, C., Joseph, A., Mehta, D., Goderstad, O., Mistri, J., Kokorelis, V. P. “*Group for Building Information Modeling*” 2011. <http://www.linkedin.com/groups/I-would-like-know-which-89922.S.66124426>. 07-Abr-2014.
- [86] <http://www.bentley.com/en-US/Products/Structural+Analysis+and+Design/ISM/>. 05-Abr-2014
- [87] Lipinski, M. “*ISM process for RAM Structural System and Revit Structure*”. <http://www.youtube.com/watch?v=crQHaeYmyjk>. 04-Apr-2014.
- [88] BuildingSMART. “*IfcMaterial*” 2013. <http://www.buildingsmart-tech.org/ifc/IFC4/final/html/schema/ifcmaterialresource/lexical/ifcmaterial.htm>. 20-Mar-2014
- [89] BuildingSMART. “*IfcRelAssociatesMaterial*”. 2013. <http://www.buildingsmart-tech.org/ifc/IFC4/final/html/index.htm>. 10-Abr-2014.
- [90] Dahlke, K. “*Determinants, Shoelace Formula*”. 2008. <http://www.mathreference.com/la-det,shoe.html>. 23-Maio-2014
- [91] BuildingSMART. “*buildingSMART data model*”. <http://www.buildingsmart-tech.org/specifications>. 25-Mar-2014.
- [92] Beal, V., Stroud, F., Shread, P. “*API - application program interface*”. <http://www.webopedia.com/TERM/A/API.html>. 24-Maio-2014.

- [93] Orenstein, D. “*QuickStudy: Application Programming Interface (API)*”. 2000.
http://www.computerworld.com/s/article/43487/Application_Programming_Interface?pageNumber=1. 24-Maio-2014.
- [94] Proffitt, B. “*What APIs Are And Why They’re Important*”. 2013.
<http://readwrite.com/2013/09/19/api-defined#awesm=~oF6lpvCBg8xo0V>. 24-Maio-2014.
- [95] Krijnen, T. “*IfcOpenShell*,” 2014. <https://github.com/aothms/IfcOpenShell>. 17-Mar-2014.
- [96] Pinto, M. L. “*Bases de Dados e Organização da Informação*”. ASA, 1996.
- [97] Pilgrim, M. “*Dive Into Python*”. 2014.

ANEXOS

A1

API'S PARA MANIPULAÇÃO DO IFC

Módulos	Sub-Módulos	Gratuito/Pago	Formato Aberto	Linguagem	Ferramentas Programáticas Associadas	Função	Formatos IFC suportados	Comentário	Última Atualização	Fontes Consultadas
IfcOpenShell	IfcMax	Gratuito	Sim	C++	Autodesk 3ds Max	Importação para Autodesk 3ds Max; Visualização do modelo em 3D.	Biblioteca para leitura e escrita de informação orientada por objetos.	Pode não reconhecer todas as componentes de um ficheiro IFC, tendo em conta que se trata de uma ferramenta sem fins lucrativos. Apesar de estar escrito em linguagem C++, existe uma transposição deste módulo para a linguagem Python.	2014	Http://ifcopenshell.org/ http://blog.ifcopenshell.org/ http://www.freecadweb.org/wiki/index.php?title=Extra_python_modules#IfcOpenShell
	IfcBlender				Blender	Importação para Blender; Visualização do modelo em 3D.				
	IfcConvert				n.a.	Exportação de IFC para: STEP (.stp); IGES (.igs); Collada Digital Asset Exchange (.dae); Wavefront OBJ (obj.). Apenas suporta IFC-SPF, i.e. o formato *.ifc.				
IfcPlusPlus (IFC++)	IfcPlusPlus	Gratuito	Sim	C++	n.a.	API em C++ baseada na informação do modelo; Biblioteca para leitura e escrita de informação orientada por objetos;	IFC 4	São ainda em número limitado as ferramentas programáticas que suportam o IFC4. Tal situação é, contudo, resolvida através da última extensão apresentada. Está escrito em linguagem C++. Qualquer alteração no código por terceiros deve ser comunicada e as alterações publicadas.	2014	https://code.google.com/p/ifcplusplus/ http://www.ifcplusplus.com/
	IfcGeometryConverter				OpenSceneGraph	Suplemento para OpenSceneGraph; Biblioteca para leitura e escrita de informação orientada por objetos;				
	IfcPlusPlusViewer				Qt OpenSceneGraph	Visualização do modelo em 3D.				
	IfcPlusPlusExtender				n.a.	Extensão para IfcPlusPlus de modo a suportar outros esquemas do IFC.				
Ifc-dotnet	-	Gratuito	Sim	C#	n.a.	Biblioteca para leitura e escrita de informação orientada por objetos. Importação de dados; Exportação de STEP e IFC-XML para .NET;	IFC 2x3	O projeto não obteve mais atualizações a partir de 2012, estando ainda na versão alpha. Por este motivo, não existem ficheiros compilados para respetivo descarregamento.	2012	https://code.google.com/p/ifc-dotnet/
ST-Developer	stdev.jar	Pago	Não	Java	n.a.	API em Java baseada na informação STEP contida no modelo para leitura e escrita no formato STEP Part 21.	IFC 2x2 IFC 2x3 IFC 4	-	2014	http://www.steptools.com/support/stdev_docs/index.html
	ROSE Library			C++	n.a.	API em C++ que utiliza classes geradas a partir do esquema EXPRESS.				
	SDAI C			C	n.a.	API em C para o esquema EXPRESS, incluindo portanto a linguagem STEP.				
	Workingset			C++	n.a.	Extensão para Rose Library de modo a suportar ficheiros "pesados". Esta aplicação apenas carrega a informação relevante executando um filtro no documento de texto.				

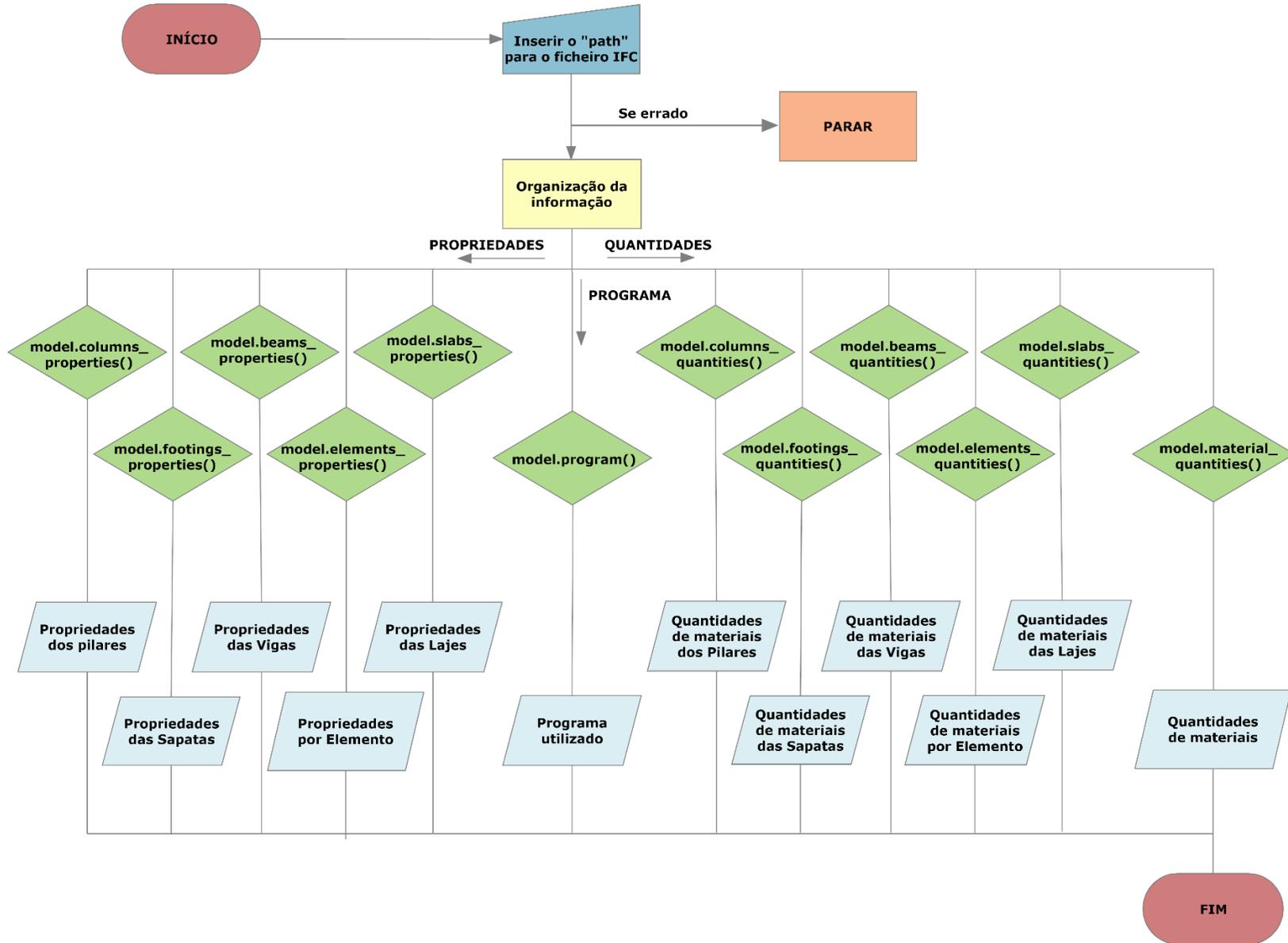
JSDAI	-	Gratuito ou Pago	Sim	Java	n.a.	API em Java para o esquema EXPRESS, direccionada para a leitura e escrita de informação orientada por objetos.	n.d.	O software pode ser editado (open source), no entanto se posteriormente for utilizado para fins comerciais, então será necessário adquirir uma licença comercial. Existe uma transposição deste módulo para a linguagem Python, atualizada até 2013.	2011	http://www.jsdai.net/ http://sourceforge.net/projects/pysdai/
IFC-SDK	-	Gratuito	Sim	C++	n.a.	API em C++ para leitura, escrita e manipulação de informação orientada a objetos; Contém uma biblioteca STEP e uma IFC com todas as entidades.	IFC 2x3	É possível compilar todas as entidades apresentadas no IFC 2x3. Tem como próximo objetivo a migração para o esquema IFC4.	2013	https://joinup.ec.europa.eu/software/ifc-sdk/description http://cad-3d.blogspot.be/2011/08/ifc-sdk-open-source-ifc-2x3-library.html
IFCsvr ActiveX Component	-	Gratuito	Não	Visual Basic	n.a.	Importação de dados; Exportação de dados para STEP Part21 e XML; Leitura e escrita de informação orientada por objetos com Visual Basic.	IFC 2x2 IFC 2x3	Tem como próximo objetivo a migração para IFC4.	2013	https://groups.yahoo.com/neo/groups/ifcsvr-users/info http://cic.vtt.fi/projects/ifcsvr/ifcsvrr200/default.html
IFC Tools Project	-	Gratuito ou Pago	Sim	Java	IFC WebGL Viewer	Visualização do modelo em 3D: API em Java para o esquema EXPRESS, direccionada para a leitura e escrita de informação orientada por objetos.	IFC2x3 IFC4	Este projeto é a "continuação" de um projeto de IFC, intitulado Open IFC Tools. O seu uso é apenas gratuito para fins de investigação.	2013	http://www.ifctoolsproject.com/
BSPRO	IfcLib	n.d	Sim	C++	n.a.	Bibliotecas para leitura e escrita de informações orientadas por objetos.	n.d.	Apresenta soluções para a interoperabilidade da informação entre algumas ferramentas computacionais a partir das quais se gerem modelos em IFC.	n.d.	http://www.granlund.fi/en/software/bspro/
	BSPROLib			C#	n.a.	Transposição da biblioteca para C# and Visual Basic.NET.				
	IfcConsole			n.d.	VisualStudio2005	Contém as funções básicas do módulo IfcLib aplicado a Visual Basic 2005.				
	IfcSpaceBoundary			n.d.	VisualStudio2005	Contém apenas as funcionalidades da IfcLib associadas à entidade IfcSpaceBoundary.				
	sb.arx			n.d.	AutoCAD 2010	Visualizador do IfcSpaceBoundary para AutoCAD.				
ECCO Toolkit	-	Gratuito para testar	Não	Java, C/C++ Perl Tcl/TK C#	n.a.	Biblioteca para leitura e escrita de informação orientada por objetos; Permite a verificação da consistência da informação do modelo; Exportação de IFC para: XML; STEP Parts 21, 25 e 28..	n.d.	Licença gratuita para 30 dias. Apresenta soluções para a interoperabilidade da informação entre algumas ferramentas computacionais.	2013	https://www.pdtec.de/pdtec/products/ecco_toolkit.aspx https://www.pdtec.de/pdtec/support/downloads/ecco/ecco-article.pdf

IFC Engine DLL	-	Pago	Sim (excepto a biblioteca DLL)	C++	IFC Viewer	Biblioteca para leitura e escrita de informação orientada por objetos; Vizualização do modelo em 3D; Exportação a partir de IFC-XML.	IFC 2x3 IFC 4	Apesar de estar definido como software pago, realça-se que a componente desta aplicação relativa à visualização do modelo em 3D é gratuita.	2013	http://rdf.bg/ifcengine.dll/product/ifcdll.html http://www.ifcbrowser.com/
----------------	---	------	-----------------------------------	-----	------------	--	------------------	---	------	--

LEGENDA	
1	API - <i>Application Programming Interface</i>
2	EPM - <i>Exchange of Product model</i>
3	n.d. - não definido
4	Alpha - Primeira versão do programa
5	n.a. - não aplicável

A2

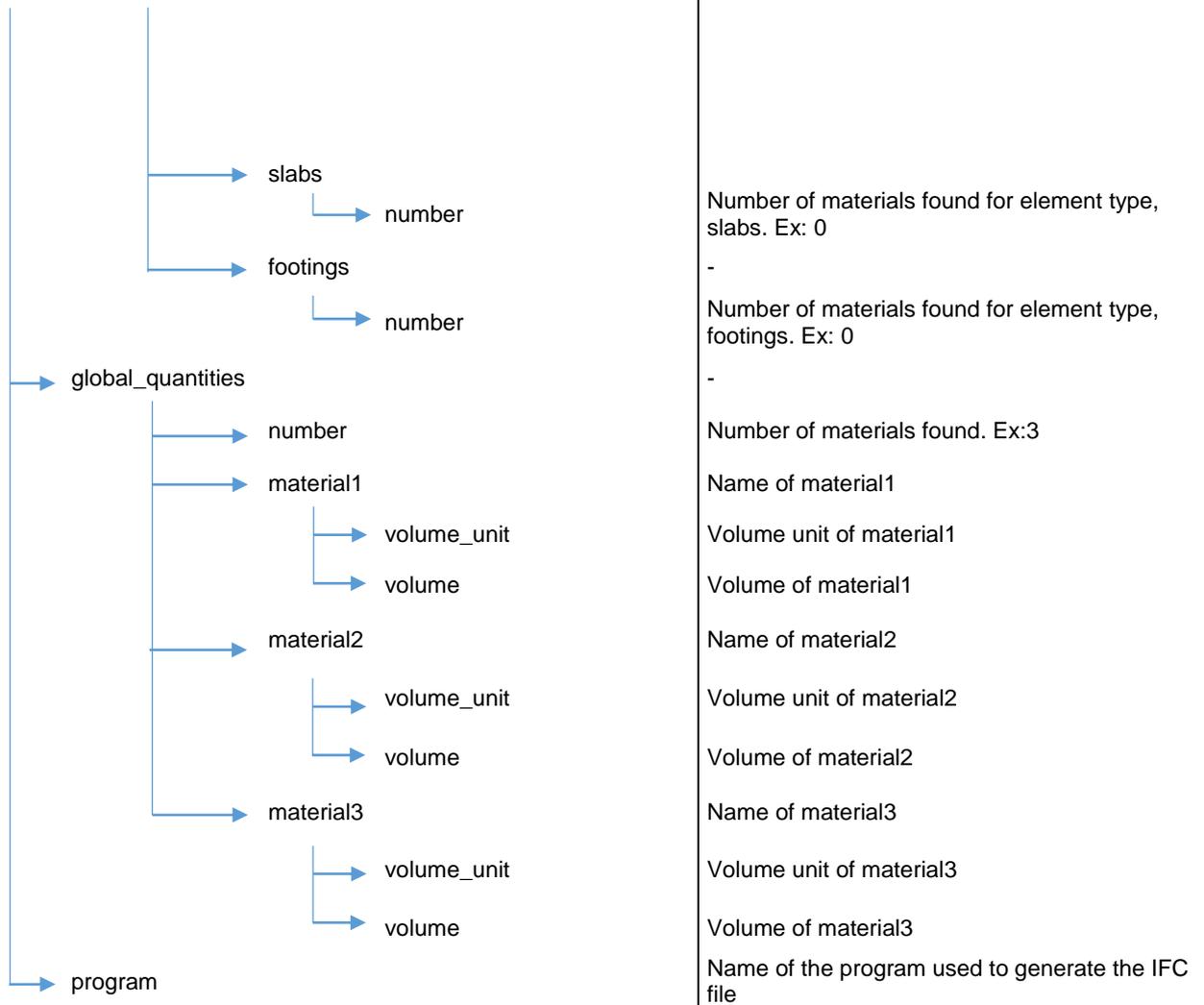
FLUXOGRAMA DO PROGRAMA PARA OBTENÇÃO AUTOMÁTICA DE QUANTIDADES



A3

HIERARQUIA ESTRUTURAL DO XML PARA FILTRAGEM DE DADOS

HIERARCHY OF INFORMATION	RETURNS
result	-
<ul style="list-style-type: none"> → header 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> → Information 	General information about the file created.
<ul style="list-style-type: none"> → body 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> → properties 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → number 	Number of elements by element type. Ex: '2 Columns', '1 Beams', '0 Slabs', '0 Footings'
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → column1 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → element_name 	Name of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → material 	Material of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → length_unit 	Length unit of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → dimension_X 	Dimension X of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → dimension_Y 	Dimension Y of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → depth 	Depth of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → area_unit 	Area unit of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → area 	Area of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume_unit 	Volume unit of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume 	Volume of column1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> → column2 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → "idem" 	"idem"
<ul style="list-style-type: none"> <ul style="list-style-type: none"> → beam1 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → "idem" 	"idem"
<ul style="list-style-type: none"> → quantities 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> → columns 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → number 	Number of materials found for element type, columns. Ex: 2
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → material1 	Name of material1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume_unit 	Volume unit of material1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume 	Volume of material1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → material2 	Name of material2
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume_unit 	Volume unit of material2
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume 	Volume of material2
<ul style="list-style-type: none"> <ul style="list-style-type: none"> → beams 	-
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → number 	Number of materials found for element type, beams. Ex: 1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → material1 	Name of material1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume_unit 	Volume unit of material1
<ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> → volume 	Volume of material1



A4

**APLICAÇÃO DO XML AO EXCEL:
COM OPÇÃO DE FILTRO DE DADOS**

No presente Anexo, apresentam-se as várias folhas de cálculo que compõem a ferramenta criada no *Microsoft Excel*. Refere-se que o separador “*Information*” não é aqui apresentado, uma vez que é referente à hierarquia estrutural do XML, a qual é exposta no Anexo 3.

	A	B	C	D
1				
2				Comment
3		Code	<result>	Insert in the cell on the left the output generated throw the Webservice.
4		header	Joao Sa; MIEC FEUP. 2013/2014	General Information about the author and the scope of this work.
5		program	Autodesk Revit 2014 (ENU)	Name of the program where the IFC file was generated from.
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				

<	>	Code	Properties	Quantities by element type	Global Quantities	Information	+
---	---	------	------------	----------------------------	-------------------	-------------	---

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2					Comment								
3			number	'4 Columns', '4 Beams', '1 Slabs', '4 Footings'	In the cell on the left it's specified the number of existing elements by element type.								
4													
5			ifc_entity	element_name	material	length_unit	dimension_X	dimension_Y	depth	area_unit	area	volume_unit	volume
6		column1	IFCCOLUMN	Concrete-Square-Column(1):25 x 25:158880	Concrete - Cast-in-Place Concrete	Meters	0.25	0.25	3.0	m2	0.1	m3	0.2
7		column2	IFCCOLUMN	Concrete-Square-Column(1):25 x 25:158882	Concrete - Cast-in-Place Concrete	Meters	0.25	0.25	3.0	m2	0.1	m3	0.2
8		column3	IFCCOLUMN	Concrete-Square-Column(1):25 x 25:158884	Concrete - Cast-in-Place Concrete	Meters	0.25	0.25	3.0	m2	0.1	m3	0.2
9		column4	IFCCOLUMN	Concrete-Square-Column(1):25 x 25:158886	Concrete - Cast-in-Place Concrete	Meters	0.25	0.25	3.0	m2	0.1	m3	0.2
10		beam1	IFCBEAM	M_Concrete-Rectangular Beam existente:25 x 40:159641	Concrete - Precast Concrete	Meters	0.4	0.25	4.75	m2	0.1	m3	0.5
11		beam2	IFCBEAM	M_Concrete-Rectangular Beam existente:25 x 40:159667	Concrete - Precast Concrete	Meters	0.4	0.25	4.75	m2	0.1	m3	0.5
12		beam3	IFCBEAM	M_Concrete-Rectangular Beam existente:25 x 40:159690	Concrete - Precast Concrete	Meters	0.4	0.25	4.75	m2	0.1	m3	0.5
13		beam4	IFCBEAM	M_Concrete-Rectangular Beam existente:25 x 40:159706	Concrete - Precast Concrete	Meters	0.4	0.25	4.75	m2	0.1	m3	0.5
14		slab1	IFCSLAB	Foundation Slab:30:159805	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	Meters	4.75	4.75	0.3	m2	22.6	m3	6.8
15		footing1	IFCFOOTING	M_Footing-Rectangular:80 x 80 x 40:159143	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	Meters	0.8	0.8	0.4	m2	0.6	m3	0.3
16		footing2	IFCFOOTING	M_Footing-Rectangular:80 x 80 x 40:159145	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	Meters	0.8	0.8	0.4	m2	0.6	m3	0.3
17		footing3	IFCFOOTING	M_Footing-Rectangular:80 x 80 x 40:159147	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	Meters	0.8	0.8	0.4	m2	0.6	m3	0.3
18		footing4	IFCFOOTING	M_Footing-Rectangular:80 x 80 x 40:159149	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	Meters	0.8	0.8	0.4	m2	0.6	m3	0.3
19		Comment	Representative entity in IFC file.	Name of the element in the model.	Name of it's material.	Properties of each element.							
20													
21													
22													
23													
24													
25													
26													
27													
28													
29													
30													
31													
32													
33													
34													
35													
36													
37													
38													
39													
40													
41													
42													

All the elements found are represented by its element type (i.e. "column", "beam", "slab" or "footing"). Then, to this representation, its added a number, in crescent order from 1 to the number calculated in cell "number", for each element type. Ex: If there are 4 Columns then we have "column1", "column2", "column3" and "column4".

Consequently, to access the data of each column he path needs to be changed accordingly to the element in analysis. The same applies for each element type.

	A	B	C	D	E	F	G	H	I	J
1										
2			number_materials	material1	volume_unit	volume		material2	volume_unit	volume
3		columns	1	Concrete - Cast-in-Place Concrete	m3	0.8		#VALUE!	#VALUE!	#VALUE!
4		beams	1	Concrete - Precast Concrete	m3	2.0		#VALUE!	#VALUE!	#VALUE!
5		slabs	1	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	m3	6.8		#VALUE!	#VALUE!	#VALUE!
6		footings	1	Concrete - Cast In Situ - Bet\X2\00E3\X0\o	m3	1.2		#VALUE!	#VALUE!	#VALUE!
7		Comment	Returns the number of materials found by element type.	If any material is found it will be specified the name of "material1" in the cells below for each element type. Per example, if columns have 2 different materials (i.e. "material1"; "material2") it's necessary to change the original path to: "=FILTERXML(Code!\$C\$3;"result/body/quantities/columns/material2")" to obtain the name of "material2".	The same applies here.			This is an example on how to obtain the properties of "material2" if found.		
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										

A5

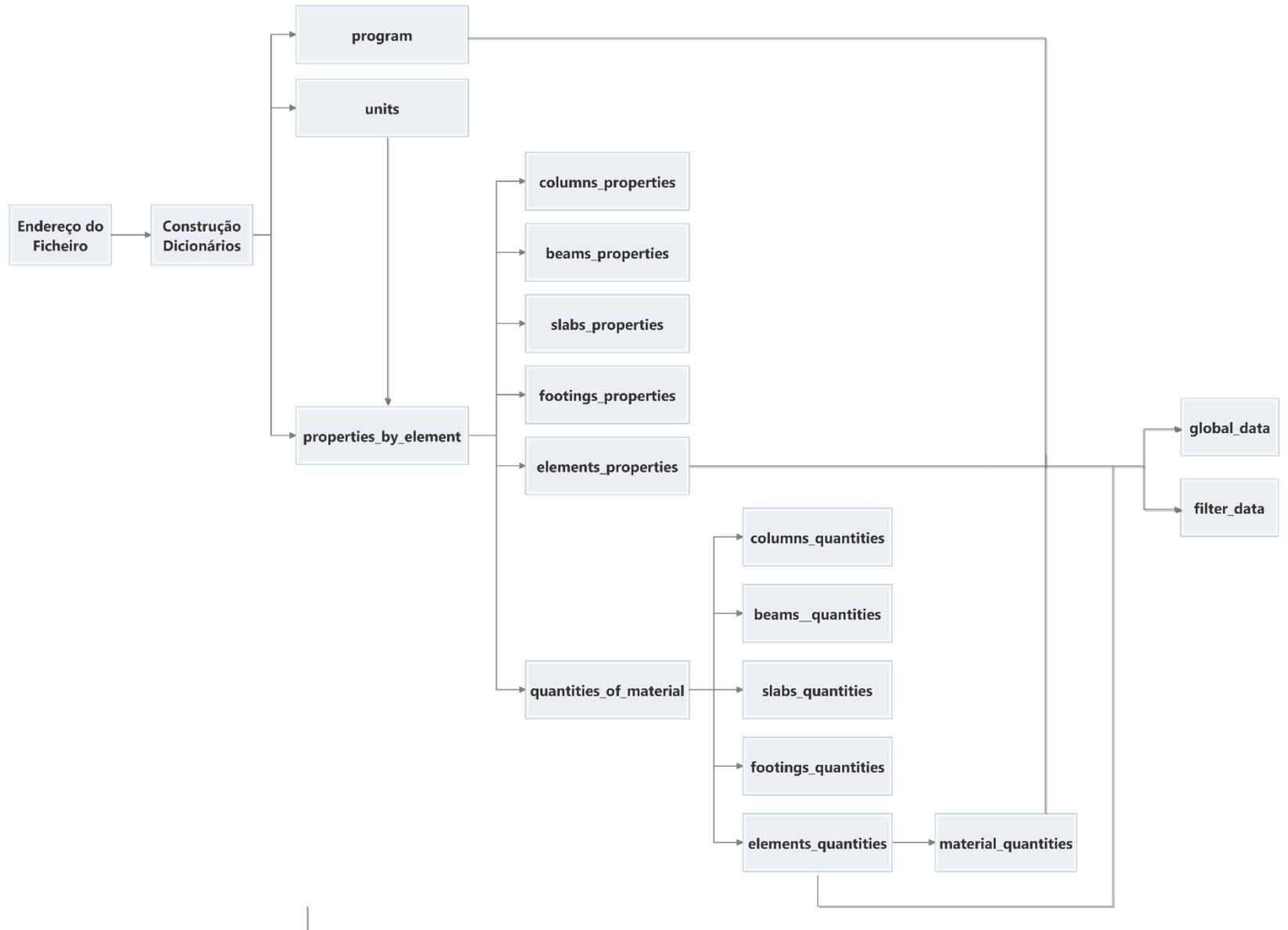
APLICAÇÃO DO XML AO EXCEL: DADOS GLOBAIS

Importação do ficheiro *.xml, contendo o resultado do método *model.global_data()*, para o *Microsoft Excel*. Utilização do modelo IFC do *Tekla Structures*:

dictionary	element_type	entity	name	material	length_unit	dimension_X	dimension_Y	depth	area_unit	area	volume_unit	volume	info	program
Element Properties	Beams	IFCBEAM	VIGA	CONCRETE/C45/55	Millimeters	No data	No data	4750	m2	6,4	m3	0,5		
Element Properties	Beams	IFCBEAM	VIGA	CONCRETE/C45/55	Millimeters	No data	No data	4750	m2	6,4	m3	0,5		
Element Properties	Beams	IFCBEAM	VIGA	CONCRETE/C45/55	Millimeters	No data	No data	4750	m2	6,4	m3	0,5		
Element Properties	Beams	IFCBEAM	VIGA	CONCRETE/C45/55	Millimeters	No data	No data	4750	m2	6,4	m3	0,5		
Element Properties	Slabs	IFCSLAB	LAJE BETAO	CONCRETE/C45/55	Millimeters	No data	No data	200	m2	22,6	m3	4,5		
Element Properties	Columns	IFCCOLUMN	PILAR	CONCRETE/C45/55	Millimeters	No data	No data	3000	m2	3,1	m3	0,2		
Element Properties	Columns	IFCCOLUMN	PILAR	CONCRETE/C45/55	Millimeters	No data	No data	3000	m2	3,1	m3	0,2		
Element Properties	Columns	IFCCOLUMN	PILAR	CONCRETE/C45/55	Millimeters	No data	No data	3000	m2	3,1	m3	0,2		
Element Properties	Footings												Or this type of element doesn't exist, or it wasn't possible to determine it's properties.	
Element Properties	Footings												Or this type of element doesn't exist, or it wasn't possible to determine it's properties.	
Element Properties	Footings												Or this type of element doesn't exist, or it wasn't possible to determine it's properties.	
Element Properties	Footings												Or this type of element doesn't exist, or it wasn't possible to determine it's properties.	
Elements Quantities	Beams			CONCRETE/C45/55							m3	2		
Elements Quantities	Slabs			CONCRETE/C45/55							m3	4,5		
Elements Quantities	Columns			CONCRETE/C45/55							m3	0,8		
Elements Quantities	Footings												Or this type of element doesn't exist, or it wasn't possible to determine it's properties.	
Material Quantities				CONCRETE/C45/55							m3	7,3		
Program														Tekla Structures Educational

A6

ESTRUTURA GLOBAL DO PROGRAMA PARA APLICAÇÃO AO WEB SERVICE



MÉTODOS DA CLASSE IFC

A7

PROGRAMAÇÃO EM *PYTHON* PARA A OBTENÇÃO DE QUANTIDADES VIA IFC

"""

Program to calculate quantities and elements properties through IFC files.

Module author: Joao Sa <ec08213@fe.up.pt>

"""

```
import re
import requests
```

class IFC(object):

"""If the path introduced is correct this class reads the file, if not it will ask for a new path. After this step, the information within the IFC file is converted into two different dictionaries.

While the first dictionary has as keys the Unique Identifiers and as values the corresponding entities and attributes, the second has as keys all the entities that form the file and as values all the correspondent Unique Identifiers.

Per example:

- First dictionary:

```
arranged_ifc = {'#1009' : ['IFCPROPERTYSET'], ['18KVJfJGL26B_mTRk$Y$kS', '#41',
                                'Pset_BeamCommon', '$', ['#809', '#322']]]}
```

- Second dictionary:

```
list_of_entities = {'IFCRELAGGREGATES' : ['#1153', '#1157', '#1161']}
```

With these dictionaries, the information can be easily manipulated.

Args:

path (str): The path to your IFC file.

Returns:

```
arranged_ifc (dict)
list_of_entities (dict)
```

"""

def __init__(self, file_adress, list_of_elements):

```
self.file_adress = file_adress
self.list_of_elements = list_of_elements
```

```
arranged_ifc = {}
list_of_entities = {}
```

try:

```
read_file = requests.get(self.file_adress).content
```

```

find_data = read_file.find('DATA;')
read_file = read_file [find_data + len('DATA;\n'):]
find_endsec = read_file.find('ENDSEC;')
read_file = read_file [:find_endsec]

i=0
while i < len(read_file):
    if read_file[i] == '#':
        start=i
        i+=1
    elif read_file[i] == '=':
        end=i
        key=read_file[start:end]
        aranged_ifc[key]=[]

        if read_file[i+1] == ' ':
            i=i+2
            start=i
        else:
            i+=1
            start=i
    elif read_file[i] == '(':
        end=i
        entity=read_file[start:end]
        aranged_ifc[key].append(entity)
        if entity not in list_of_entities:
            list_of_entities[entity] = [key]
        else:
            list_of_entities[entity].append(key)

        start=end+1
        end=read_file[start:].find(';')+start-1
        read_file_temp=read_file[start:end]
        read_file_temp=read_file_temp.replace(""," ")
        read_file_temp=read_file_temp.replace("\n","")
        read_file_temp=read_file_temp.replace("","")

        aranged_ifc[key].append([read_file_temp])
        i=end+2
    else:

```

```

        i+=1

    for key in aranged_ifc:
        aranged_ifc[key][1] = re.findall("\([^\)]*\)|[^\,]+'", aranged_ifc[key][1][0])
        for i in range(len(aranged_ifc[key][1])):
            if aranged_ifc[key][1][i].startswith('#'):
                aranged_ifc[key][1][i] = aranged_ifc[key][1][i][1:-1].split(',')
            else:
                pass

    self.aranged_ifc = aranged_ifc
    self.list_of_entities = list_of_entities

except IOError:
    print 'Error, can\'t find file or read data. Please try again.'

def units(self):
    """This method determines the length unit presented in the IFC file. Depending on the information of
    this entity, it will store the unit as a string, as well as the correspondent volume unit. This volume unit is
    obtained considering the multiplication of values with the length unit obtained.

    It's known that the IFC file may contain volume information and, consequently, the correspondent
    volume unit, may not correspond to the real volume unit in the model. For this reason, the real volume unit
    is also determined because it's use can be necessary in the following methods. The same happens with area
    unit and correspondent area unit.

    Args:
        path (str): The path to your IFC file.

    Returns:
        length_unit (str)
        correspondent_volume_unit (str)
        corresponded_area_unit (str)
        volume_unit (str)
        area_unit (str)
    """

    IFCUNITASSIGNMENT = self.list_of_entities['IFCUNITASSIGNMENT']
    volume_unit = ''

    for value in IFCUNITASSIGNMENT: # {'key':'value'} value in self.list_of_entities is the key in
    self.aranged_ifc
        for i in range(len(self.aranged_ifc[value][1][0])-5):
            if self.aranged_ifc[self.aranged_ifc[value][1][0][i]][1][1] == '.LENGTHUNIT.':

```

```

prefix = self.aranged_ifc[self.aranged_ifc[value][1][0][i]][1][2]
if prefix == '$':
    length_unit = 'Meters'
    correspondent_volume_unit = 'm3'
    correspondent_area_unit = 'm2'
elif prefix == '.CENTI.':
    length_unit = 'Centimeters'
    correspondent_volume_unit = 'cm3'
    correspondent_area_unit = 'cm2'
elif prefix == '.DECI.':
    length_unit = 'Decimeters'
    correspondent_area_unit = 'dm2'
    correspondent_volume_unit = 'dm3'
elif prefix == '.MILLI.':
    length_unit = 'Millimeters'
    correspondent_volume_unit = 'mm3'
    correspondent_area_unit = 'mm2'
else:
    length_unit = 'Not recognized'
    correspondent_volume_unit = 'Not recognized'
    correspondent_area_unit = 'Not recognized'
elif self.aranged_ifc[self.aranged_ifc[value][1][0][i]][1][1] == '.VOLUMEUNIT.':
    prefix = self.aranged_ifc[self.aranged_ifc[value][1][0][i]][1][2]
    if prefix == '$':
        volume_unit = 'm3'
    else:
        volume_unit = 'Not recognized'
if self.aranged_ifc[self.aranged_ifc[value][1][0][i]][1][1] == '.AREAUNIT.':
    prefix = self.aranged_ifc[self.aranged_ifc[value][1][0][i]][1][2]
    if prefix == '$':
        area_unit = 'm2'
    else:
        area_unit = 'Not recognized'

return length_unit, correspondent_volume_unit, volume_unit, correspondent_area_unit, area_unit

```

def program(self):

"""This method searches for the attribute of the entity where the name of the program used to export the IFC file is specified and then returns it as a string.

Args:

path (str): The path to your IFC file.

Returns:

program_name (str)

"""

```
program = { 'Program' : [] }
IFCAPPLICATION = self.list_of_entities['IFCAPPLICATION']
program_name = self.aranged_ifc[IFCAPPLICATION[0]][1][2]
program['Program'] = program_name
return program
```

def properties_by_element(self, list_of_elements):

""" This method searches for all the properties that are relevant for each structural element specified in the argument ("list_of_elements") and returns a dictionary containing them.

The "list_of_elements" is yet limited to columns, beams, slabs and footings.

The dictionary presents the information in the following format:

```
properties = { 'Type of element' : [[representative entity, element name, material, length unit,
dimension X, dimension Y, depth, area unit, area, volume unit, volume], [...]], [...] }
```

Each sublist represents one single element.

Args:

path (str): The path to your IFC file.

list_of_elements (list): List of strings. Each string corresponds to the representative IFC entity of the structural member.

Returns:

properties (dict)

"""

```
length_unit, correspondent_volume_unit, volume_unit, correspondent_area_unit, area_unit =
self.units()
```

```
elements = []
```

```
for element in list_of_elements:
```

```
    if element in self.list_of_entities:
```

```
        [elements.append(self.list_of_entities[element][i]) for i in
range(len(self.list_of_entities[element]))]
```

```
if 'IFCEXTRUDEDAREASOLID' in self.list_of_entities:
```

```
    IFCPRODUCTDEFINITIONSHAPE = []
```

```
    IFCSHAPEREPRESENTATION = []
```

```
    IFCMAPPEDITEM = []
```

```
    IFCREPRESENTATIONMAP = []
```

```
    IFCSHAPEREPRESENTATION1 = []
```

```
    IFCEXTRUDEDAREASOLID1 = []
```

```
    IFCEXTRUDEDAREASOLID2 = []
```

```
    IFCRECTANGLEPROFILEDEF = []
```

```
    IFCPOLYLINE = []
```

```
    IFCCARTESIANPOINT = []
```

```
    IFCARBITRARYCLOSEDPROFILEDEF = []
```

```
representative_entity = []
```

```
material = []
```

```
element_name = []
```

```
depth = []
```

```
dimension_X = []
```

```
dimension_Y = []
```

```
relating_material = []
```

```
coordinates = []
```

```
area = []
```

```
for element in elements:
```

```
    element_name.append(self.aranged_ifc[element][1][2])
```

```
    representative_entity.append(self.aranged_ifc[element][0])
```

```
    if self.aranged_ifc[self.aranged_ifc[element][1][6]][0] ==
'IFCPRODUCTDEFINITIONSHAPE':
```

```
        IFCPRODUCTDEFINITIONSHAPE.append(self.aranged_ifc[element][1][6])
```

```

for element in elements:
    elements_with_same_material =
[self.aranged_ifc[self.list_of_entities['IFCRELASSOCIATESMATERIAL'][i]][1][4] for i in
range(len(self.list_of_entities['IFCRELASSOCIATESMATERIAL']))]
    for i in range(len(elements_with_same_material)):
        if element in elements_with_same_material[i]:

relating_material.append(self.aranged_ifc[self.list_of_entities['IFCRELASSOCIATESMATERIAL'][i]][1]
[5])

    else:
        pass

for entity in relating_material:
    if self.aranged_ifc[entity][0] == 'IFCMATERIAL':
        material.append(self.aranged_ifc[entity][1][0])
    elif self.aranged_ifc[entity][0] == 'IFCMATERIALLAYERSETUSAGE':
        material.append(str([self.aranged_ifc[self.aranged_ifc[self.aranged_ifc[self.aranged_ifc
[entity][1][0]][1][0][i]][1][0]][1][0] for i in
range(len(self.aranged_ifc[self.aranged_ifc[entity][1][0]][1][0]))][2:-2])
    elif self.aranged_ifc[entity][0] == 'IFCMATERIALAYERSET':

material.append([self.aranged_ifc[self.aranged_ifc[self.aranged_ifc[entity][1][0][i]][1][0][1][0]] for i in
range(len(self.aranged_ifc[entity][1][0]))])
    elif self.aranged_ifc[entity][0] == 'IFCMATERIALIST':
        material.append([self.aranged_ifc[self.aranged_ifc[entity][1][0][i]][1][0] for i in
range(len(self.aranged_ifc[entity][1][0]))])

for value in IFCPRODUCTDEFINITIONSHAPE:
    for i in range(len(self.aranged_ifc[value][1][2])):
        if self.aranged_ifc[self.aranged_ifc[value][1][2][i]][0] == 'IFCSHAPEREPRESENTATION'
and self.aranged_ifc[self.aranged_ifc[value][1][2][i]][1][1] == 'Body':
            IFCSHAPEREPRESENTATION.append(self.aranged_ifc[value][1][2][i])

for value in IFCSHAPEREPRESENTATION:
    for i in range(len(self.aranged_ifc[value][1][3])):
        if self.aranged_ifc[self.aranged_ifc[value][1][3][i]][0] == 'IFCMAPPEDITEM':
            IFCMAPPEDITEM.append(self.aranged_ifc[value][1][3][i])
            IFCEXTRUDEDAREASOLID1.append('IFCMAPPEDITEM')
        elif self.aranged_ifc[self.aranged_ifc[value][1][3][i]][0] == 'IFCEXTRUDEDAREASOLID':
            IFCEXTRUDEDAREASOLID1.append(self.aranged_ifc[value][1][3][i])
            IFCMAPPEDITEM.append('IFCEXTRUDEDAREASOLID')

```

```

if list(set(IFCMAPPEDITEM)) != ['IFCEXTRUDEDAREASOLID']:
    i=0
    while i < len(IFCMAPPEDITEM):
        if IFCMAPPEDITEM[i] != 'IFCEXTRUDEDAREASOLID':
            if self.aranged_ifc[self.aranged_ifc[IFCMAPPEDITEM[i]][1][0]][0] ==
'IFCREPRESENTATIONMAP':
                IFCREPRESENTATIONMAP.append(self.aranged_ifc[IFCMAPPEDITEM[i]][1][0])
                i+=1

            elif IFCMAPPEDITEM[i] == 'IFCEXTRUDEDAREASOLID':
                IFCREPRESENTATIONMAP.append('IFCEXTRUDEDAREASOLID')
                i+=1

    i=0
    while i < len(IFCREPRESENTATIONMAP):
        if IFCREPRESENTATIONMAP[i] != 'IFCEXTRUDEDAREASOLID':
            if self.aranged_ifc[self.aranged_ifc[IFCREPRESENTATIONMAP[i]][1][1]][0] ==
'IFCSHAPEREPRESENTATION':
                IFCSHAPEREPRESENTATION1.append(self.aranged_ifc[IFCREPRESENTATIONMAP[i]][1][1])
                i+=1
            elif IFCREPRESENTATIONMAP[i] == 'IFCEXTRUDEDAREASOLID':
                IFCSHAPEREPRESENTATION1.append('IFCEXTRUDEDAREASOLID')
                i+=1

    i=0
    while i < len(IFCSHAPEREPRESENTATION1):
        if IFCSHAPEREPRESENTATION1[i] != 'IFCEXTRUDEDAREASOLID':
            for j in range(len(self.aranged_ifc[IFCSHAPEREPRESENTATION1[i]][1][3])):
                if self.aranged_ifc[self.aranged_ifc[IFCSHAPEREPRESENTATION1[i]][1][3][j]][0] ==
'IFCEXTRUDEDAREASOLID':
                    IFCEXTRUDEDAREASOLID2.append(self.aranged_ifc[IFCSHAPEREPRESENTATION1[i]][1][3][j])
                    i+=1

            elif IFCSHAPEREPRESENTATION1[i] == 'IFCEXTRUDEDAREASOLID':
                IFCEXTRUDEDAREASOLID2.append('IFCEXTRUDEDAREASOLID')
                i+=1

    i=0
    while i < len(IFCEXTRUDEDAREASOLID1):

```

```

if IFCEXTRUDEDAREASOLID1[i] == 'IFCMAPPEDITEM':
    IFCEXTRUDEDAREASOLID1[i] = IFCEXTRUDEDAREASOLID2[i]
    i+=1
else:
    i+=1

for value in IFCEXTRUDEDAREASOLID1:
    if value != 'IFCMAPPEDITEAM':
        depth.append(self.aranged_ifc[value][1][3])
        if self.aranged_ifc[self.aranged_ifc[value][1][0]][0] == 'IFCRECTANGLEPROFILEDEF':
            IFCRECTANGLEPROFILEDEF.append(self.aranged_ifc[value][1][0])
        elif self.aranged_ifc[self.aranged_ifc[value][1][0]][0] != 'IFCRECTANGLEPROFILEDEF':
            IFCRECTANGLEPROFILEDEF.append('None')

for value in IFCRECTANGLEPROFILEDEF:
    if value == 'None':
        dimension_X.append(0)
        dimension_Y.append(0)
    else:
        dimension_X.append(round(float(self.aranged_ifc[value][1][3]),2))
        dimension_Y.append(round(float(self.aranged_ifc[value][1][4]),2))

for value in IFCEXTRUDEDAREASOLID1:
    if self.aranged_ifc[self.aranged_ifc[value][1][0]][0] ==
'IFCARBITRARYCLOSEDPROFILEDEF':
        IFCARBITRARYCLOSEDPROFILEDEF.append(self.aranged_ifc[value][1][0])
    elif self.aranged_ifc[self.aranged_ifc[value][1][0]][0] !=
'IFCARBITRARYCLOSEDPROFILEDEF':
        IFCARBITRARYCLOSEDPROFILEDEF.append('None')

for value in IFCARBITRARYCLOSEDPROFILEDEF:
    if value == 'None':
        IFCPOLYLINE.append('None')
    elif self.aranged_ifc[self.aranged_ifc[value][1][2]][0] == 'IFCPOLYLINE':
        IFCPOLYLINE.append(self.aranged_ifc[value][1][2])

for value in IFCPOLYLINE:
    if value == 'None':
        IFCCARTESIANPOINT.append('None')

```

```

else:
    IFCCARTESIANPOINT.append(self.arrengeo_ifc[value][1][0])

for value in IFCCARTESIANPOINT:
    if value == 'None':
        coordinates.append('None')
    else:
        coordinates.append([self.arrengeo_ifc[value[i]][1][0] for i in range(len(value))])

coordinates_rearranged = []

for group_coordinates in coordinates:
    if group_coordinates == 'None':
        coordinates_rearranged.append('None')
    else:
        new_coordinate = [tuple(float(v) for v in re.findall("[0.-9.]+", coordinate)) for coordinate in
group_coordinates]
        coordinates_rearranged.append(new_coordinate)

for group_coordinates in coordinates_rearranged:
    if group_coordinates == 'None':
        area.append(0)

    else:
        element_area = 0.5 * abs(sum(x0*y1 - x1*y0 for ((x0, y0), (x1, y1)) in zip(group_coordinates,
group_coordinates[1:] + [group_coordinates[0]])))
        area.append(round(element_area,1))

for i in range(len(area)):
    if area[i] == 0:
        area[i] = round(dimension_X[i]*dimension_Y[i],1)

depth = [round(float(i),6) for i in depth]

for i in range(len(dimension_X)):
    if dimension_X[i] != 0:
        dimension_X[i] = float(dimension_X[i])
    if dimension_Y[i] != 0:
        dimension_Y[i] = float(dimension_Y[i])

```

```
elements_description = [[representative_entity[i], element_name[i], material[i], length_unit,
float(dimension_X[i]), float(dimension_Y[i]), float(depth[i]), correspondent_area_unit, area[i],
correspondent_volume_unit] for i in range(len(dimension_X))]
```

```
for i in range(len(elements_description)):
```

```
    elements_description[i].append(float(elements_description[i][4])*float(elements_description[i][5])
*float(elements_description[i][6]))
```

```
    elements_description[i][10] = round(float(elements_description[i][10]),1)
```

```
for element in elements_description:
```

```
    if (element[4] and element[5]) == 0:
```

```
        element[4] = 'No data'
```

```
        element[5] = 'No data'
```

```
else:
```

```
    IFCELEMENTQUANTITY = []
```

```
    relating_material = []
```

```
    representative_entity = []
```

```
    material = []
```

```
    element_name = []
```

```
    depth = []
```

```
    volume = []
```

```
    area = []
```

```
for element in elements:
```

```
    element_name.append(self.aranged_ifc[element][1][2])
```

```
    representative_entity.append(self.aranged_ifc[element][0])
```

```
    for entity in self.list_of_entities['IFCRELDEFINESBYPROPERTIES']:
```

```
        if element in self.aranged_ifc[entity][1][4] and
```

```
self.aranged_ifc[self.aranged_ifc[entity][1][5]][0] == 'IFCELEMENTQUANTITY':
```

```
            IFCELEMENTQUANTITY.append(self.aranged_ifc[entity][1][5])
```

```
    elements_with_same_material =
```

```
[self.aranged_ifc[self.list_of_entities['IFCRELASSOCIATESMATERIAL'][i]][1][4] for i in
range(len(self.list_of_entities['IFCRELASSOCIATESMATERIAL']))]
```

```
for i in range(len(elements_with_same_material)):
```

```
    if element in elements_with_same_material[i]:
```

```

        relating_material.append(self.aranged_ifc[self.list_of_entities['IFCRELASSOCIATESMATERIAL'][i]][1][5])
    else:
        pass # insert "Material not associated"

for entity in IFCELEMENTQUANTITY:
    for i in range(len(self.aranged_ifc[entity][1][5])):
        if self.aranged_ifc[self.aranged_ifc[entity][1][5][i]][0] == 'IFCQUANTITYLENGTH':
            depth.append(round(float(self.aranged_ifc[self.aranged_ifc[entity][1][5][i]][1][3]),2))
        elif self.aranged_ifc[self.aranged_ifc[entity][1][5][i]][0] == 'IFCQUANTITYVOLUME':
            volume.append(round(float(self.aranged_ifc[self.aranged_ifc[entity][1][5][i]][1][3]),1))
        elif self.aranged_ifc[self.aranged_ifc[entity][1][5][i]][0] == 'IFCQUANTITYAREA':
            area.append(round(float(self.aranged_ifc[self.aranged_ifc[entity][1][5][i]][1][3]),1))

for entity in relating_material:
    if self.aranged_ifc[entity][0] == 'IFCMATERIAL':
        material.append(self.aranged_ifc[entity][1][0])
    elif self.aranged_ifc[entity][0] == 'IFCMATERIALLAYERSETUSAGE':
        material.append([self.aranged_ifc[self.aranged_ifc[self.aranged_ifc[self.aranged_ifc[entity][1][0]][1][0][i]][1][0]][1][0] for i in range(len(self.aranged_ifc[self.aranged_ifc[entity][1][0]][1][0]))])
    elif self.aranged_ifc[entity][0] == 'IFCMATERIALLAYERSET':
        material.append([self.aranged_ifc[self.aranged_ifc[self.aranged_ifc[entity][1][0][i]][1][0]][1][0] for i in range(len(self.aranged_ifc[entity][1][0]))])

material.append([self.aranged_ifc[self.aranged_ifc[self.aranged_ifc[entity][1][0][i]][1][0]][1][0] for i in range(len(self.aranged_ifc[entity][1][0]))])
    elif self.aranged_ifc[entity][0] == 'IFCMATERIALIST':
        material.append([self.aranged_ifc[self.aranged_ifc[entity][1][0][i]][1][0] for i in range(len(self.aranged_ifc[entity][1][0]))])

dimension_X = 'No data'
dimension_Y = 'No data'

elements_description = [[representative_entity[i], element_name[i], material[i], length_unit, dimension_X, dimension_Y, depth[i], area_unit, area[i], volume_unit, volume[i]] for i in range(len(depth))]

properties = {'Columns': [], 'Beams': [], 'Slabs': [], 'Footings': []}

if 'IFCCOLUMN' not in list_of_elements:
    del properties['Columns']
if 'IFCBEAM' not in list_of_elements:
    del properties['Beams']
if 'IFCSLAB' not in list_of_elements:
    del properties['Slabs']

```

```

if 'IFCFOOTING' not in list_of_elements:
    del properties['Footings']

i=0
while i < len(elements_description):
    if elements_description[i][0] == 'IFCCOLUMN':
        properties['Columns'].append(elements_description[i])
    elif elements_description[i][0] == 'IFCBEAM':
        properties['Beams'].append(elements_description[i])
    elif elements_description[i][0] == 'IFCSLAB':
        properties['Slabs'].append(elements_description[i])
    elif elements_description[i][0] == 'IFCFOOTING':
        properties['Footings'].append(elements_description[i])
    i+=1

for key in properties:
    if properties[key] == [] and [element in self.list_of_entities for element in list_of_elements]:
        i = 0
        for i in range(len(self.list_of_entities[element])):
            properties[key].append("Or this type of element doesn't exist, or it wasn't possible to determine
it's properties.")
            i += 1

return properties

```

def quantities_of_material(self,element):

""" This method determines the quantities by material associated to the structural element, specified in the argument "element". The element specified is limited to columns, beams, slabs and footings.

It returns a dictionary in the following format:

```
material = {'material_x' : ['m3', 45.002], 'material_y' : ['m3', 86.324] }
```

Args:

path (str): The path to your IFC file.

element (str): element can be equal to 'Columns', 'Beams', 'Slabs' or 'Footings'

Returns:

material (dict)

"""

```

properties = self.properties_by_element(['IFCCOLUMN','FCBEAM','FCSLAB','FCFOOTING'])

all_materials= []
material = { }
for i in range(len(properties[element])):
    if properties[element][i] != "Or this type of element doesn't exist, or it wasn't possible to determine
it's properties.":
        all_materials.append(properties[element][i][2])
        if properties[element][i][6] == 'Not recognized':
            material['Error'] = ['Not all units have been recognized and consequently it\'s impossible to
calculate all quantities associated']
            break

eliminate_duplicates_material = list(set(all_materials))

for i in range(len(properties[element])):
    if properties[element][i][9] == 'mm3':
        properties[element][i][10] = float(properties[element][i][10])*10**(-9)
    elif properties[element][i][9] == 'cm3':
        properties[element][i][10] = float(properties[element][i][10])*10**(-6)
    elif properties[element][i][9] == 'dm3':
        properties[element][i][10] = float(properties[element][i][10])*10**(-3)
    elif properties[element][i][9] == 'm3':
        pass

if material == { }:
    j=0
    sum = 0
    while j < len(eliminate_duplicates_material):
        for i in range(len(properties[element])):
            if properties[element][i][2] == eliminate_duplicates_material[j]:
                sum += round(float(properties[element][i][10]),1)
            material[eliminate_duplicates_material[j]] = ['m3', sum]
        j += 1
        sum = 0

if material == { }:
    material['Error'] = "Or this type of element doesn't exist, or it wasn't possible to determine it's
properties."
    return material

```

def columns_properties(self):

""" This method is an application of the method "properties_by_element" using as its argument ("list_of_elements"), the structural element type Columns.

```
list_of_elements = ['IFCCOLUMN']
```

Args:

path (str): The path to your IFC file.

Returns:

properties (dict)

"""

```
properties = self.properties_by_element(['IFCCOLUMN'])
```

```
return properties
```

def beams_properties(self):

""" This method is an application of the method "properties_by_element" using as its argument ("list_of_elements"), the structural element type Beams.

```
list_of_elements = ['IFCBEAM']
```

Args:

path (str): The path to your IFC file.

Returns:

properties (dict)

"""

```
properties = self.properties_by_element(['IFCBEAM'])
```

```
return properties
```

def slabs_properties(self):

""" This method is an application of the method "properties_by_element" using as its argument ("list_of_elements"), the structural element type Slabs.

```
list_of_elements = ['IFCSLAB']
```

Args:

path (str): The path to your IFC file.

Returns:

properties (dict)

"""

```
properties = self.properties_by_element(['IFCSLAB'])
return properties
```

def footings_properties(self):

""" This method is an application of the method "properties_by_element" using as its argument ("list_of_elements"), the structural element type Footings.

```
list_of_elements = ['IFCFOOTING']
```

Args:

path (str): The path to your IFC file.

Returns:

properties (dict)

"""

```
properties = self.properties_by_element(['IFCFOOTING'])
return properties
```

def elements_properties(self):

""" This method is an application of the method "properties_by_element" using as its argument ("list_of_elements"), the following structural element types:

Columns; Beams; Slabs; Footings.

```
list_of_elements = ['IFCCOLUMN', 'IFCBEAM', 'IFCSLAB', 'IFCFOOTING']
```

Args:

path (str): The path to your IFC file.

Returns:

properties (dict)

"""

```
properties = self.properties_by_element(['IFCCOLUMN', 'IFCBEAM', 'IFCSLAB', 'IFCFOOTING'])
return properties
```

def columns_quantities(self):

""" This method is an application of the method "quantities_of_material" for the structural element type Columns.

element = 'Columns'

Args:

path (str): The path to your IFC file.

Returns:

material (dict)

"""

material = self.quantities_of_material('Columns')

return material

def beams_quantities(self):

""" This method is an application of the method "quantities_of_material" for the structural element type Beams.

element = 'Beams'

Args:

path (str): The path to your IFC file.

Returns:

material (dict)

"""

material = self.quantities_of_material('Beams')

return material

def slabs_quantities(self):

""" This method is an application of the method "quantities_of_material" for the structural element type Slabs.

element = 'Slabs'

Args:

path (str): The path to your IFC file.

Returns:

material (dict)

"""

```
material = self.quantities_of_material('Slabs')
```

```
return material
```

def footings_quantities(self):

""" This method is an application of the method "quantities_of_material" for the structural element type Footings.

```
element = 'Footings'
```

Args:

path (str): The path to your IFC file.

Returns:

material (dict)

"""

```
material = self.quantities_of_material('Footings')
```

```
return material
```

def elements_quantities(self):

""" This method is an application of the method "quantities_of_material" for the structural element types:

Columns; Beams; Slabs; Footings.

Then it creates a new dictionary containing for each element referred the materials and quantities associated.

Per example:

```
elements_material = {'Columns': [['material_x', '45.002 m3'], ['material_y', '86.324 m3'],  
                                'Beams': [['material_z', '34.232 m3']], ...}
```

Args:

path (str): The path to your IFC file.

Returns:

material (dict)

"""

```
material_columns = self.quantities_of_material('Columns')
```

```
material_beams = self.quantities_of_material('Beams')
```

```
material_slabs = self.quantities_of_material('Slabs')
```

```
material_footings = self.quantities_of_material('Footings')
```

```
elements_material = {'Columns' : [], 'Beams' : [], 'Slabs' : [], 'Footings' : [] }
```

```
for key in material_columns:
```

```
    elements_material['Columns'].append([key, material_columns[key]])
```

```
for key in material_beams:
```

```
    elements_material['Beams'].append([key, material_beams[key]])
```

```
for key in material_slabs:
```

```
    elements_material['Slabs'].append([key, material_slabs[key]])
```

```
for key in material_footings:
```

```
    elements_material['Footings'].append([key, material_footings[key]])
```

```
return elements_material
```

def materials_quantities(self):

"""This method uses the dictionary "elements_quantities" to return total quantities by material (independent from element type) as a dictionary.

Per example:

```
total_materials = {'material_x' : ['m3', 100.242], 'material_y' : ['m3', 123.165]}
```

Args:

path (str): The path to your IFC file.

Returns:

total_materials (dict)

"""

```
elements_material = self.elements_quantities()
```

```

list_of_materials = []
list_of_quantities = []
for key in elements_material:
    for i in range(len(elements_material[key])):
        if elements_material[key][0][0] != 'Error':
            list_of_materials.append(elements_material[key][i][0])
            list_of_quantities.append(float(elements_material[key][i][1][1]))

quantities_temp = []
quantities = []
j = 0
while j < len(list(set(list_of_materials))):
    for i in range(len(list_of_materials)):
        if list_of_materials[i] == list(set(list_of_materials))[j]:
            quantities_temp.append(list_of_quantities[i])
    quantities.append(sum(quantities_temp))
    quantities_temp = []
    j+=1

total_materials = {}

for material, quantity in zip(set(list_of_materials), quantities):
    total_materials[material] = ['m3', round(quantity,1)]

return total_materials

```

def global_data(self):

"""This method returns a global dictionary in XML, containing four distinct dictionaries obtained by calling the following methods: elements_properties; elements_quantities; material_quantities; program. The XML structure is built to be used in Microsoft Excel with the option "From XML Data Import".

"""

```

global_dict = {}
elements_properties = self.elements_properties()
elements_quantities = self.elements_quantities()
material_quantities = self.materials_quantities()
program = self.program()
global_dict['Elements Properties'] = elements_properties
global_dict['Elements Quantities'] = elements_quantities

```

```
global_dict['Material Quantities'] = material_quantities
global_dict['Program'] = program
```

```
XML = '<global>\n<table>\n<dictionary>Element Properties</dictionary>\n'
elements = []
```

```
for key, value in dict.items(global_dict['Elements Properties']):
    elements.append(key)
```

```
i=0
```

```
for i in range(len(elements)):
```

```
    element_XML = '<data>\n<element_type>' + elements[i] + '</element_type>\n'
```

```
    XML = XML + element_XML
```

```
    for j in range(len(global_dict['Elements Properties'][elements[i]])):
```

```
        if global_dict['Elements Properties'][elements[i]][0] == "Or this type of element doesn't exist, or it wasn't possible to determine it's properties.":
```

```
            info_XML = '<item>\n<info>' + "Or this type of element doesn't exist, or it wasn't possible to determine it's properties." + '</info>\n'
```

```
            XML += info_XML + '</item>\n'
```

```
        elif global_dict['Elements Properties'][elements[i]][0] != "Or this type of element doesn't exist, or it wasn't possible to determine it's properties.":
```

```
            entity_XML = '<item>\n<entity>' + global_dict['Elements Properties'][elements[i]][j][0] + '</entity>\n'
```

```
            name_XML = '<name>' + global_dict['Elements Properties'][elements[i]][j][1] + '</name>\n'
```

```
            material_XML = '<material>' + global_dict['Elements Properties'][elements[i]][j][2] + '</material>\n'
```

```
            length_unit_XML = '<length_unit>' + global_dict['Elements Properties'][elements[i]][j][3] + '</length_unit>\n'
```

```
            dimension_X_XML = '<dimension_X>' + str(global_dict['Elements Properties'][elements[i]][j][4]) + '</dimension_X>\n'
```

```
            dimension_Y_XML = '<dimension_Y>' + str(global_dict['Elements Properties'][elements[i]][j][5]) + '</dimension_Y>\n'
```

```
            depth_XML = '<depth>' + str(global_dict['Elements Properties'][elements[i]][j][6]) + '</depth>\n'
```

```
            area_unit_XML = '<area_unit>' + global_dict['Elements Properties'][elements[i]][j][7] + '</area_unit>\n'
```

```
            area_XML = '<area>' + str(global_dict['Elements Properties'][elements[i]][j][8]) + '</area>\n'
```

```
            volume_unit_XML = '<volume_unit>' + global_dict['Elements Properties'][elements[i]][j][9] + '</volume_unit>\n'
```

```
            volume_XML = '<volume>' + str(global_dict['Elements Properties'][elements[i]][j][10]) + '</volume>\n'
```

```

XML += entity_XML + name_XML + material_XML + length_unit_XML +
dimension_X_XML + dimension_Y_XML + depth_XML + area_unit_XML + area_XML +
volume_unit_XML + volume_XML + '</item>\n'
    j+=1
XML += '</data>\n'

    i+=1
XML += '</table>\n<table>\n<dictionary>Elements Quantities</dictionary>\n'

i=0
for i in range(len(elements)):

    element_XML = '<data>\n<element_type>' + elements[i] + '</element_type>\n'
    XML = XML + element_XML
    for j in range(len(global_dict['Elements Quantities'][elements[i]])):
        if global_dict['Elements Quantities'][elements[i]][0][1] == "Or this type of element doesn't exist,
or it wasn't possible to determine it's properties.":
            info_XML = '<item>\n<info>' + "Or this type of element doesn't exist, or it wasn't possible to
determine it's properties." + '</info>\n'

            XML += info_XML + '</item>\n'

        elif global_dict['Elements Quantities'][elements[i]][0][1] != "Or this type of element doesn't exist,
or it wasn't possible to determine it's properties.":
            material_XML = '<item>\n<material>' + global_dict['Elements Quantities'][elements[i]][j][0] +
'</material>\n'
            volume_unit_XML = '<volume_unit>' + global_dict['Elements
Quantities'][elements[i]][j][1][0] + '</volume_unit>\n'
            volume_XML = '<volume>' + str(global_dict['Elements Quantities'][elements[i]][j][1][1]) +
'</volume>\n'

            XML += material_XML + volume_unit_XML + volume_XML + '</item>\n'
        j+=1
    XML += '</data>\n'

    i+=1

XML += '</table>\n<table>\n<dictionary>Material Quantities</dictionary>\n'

materials = []

for key, value in dict.items(global_dict['Material Quantities']):

```

```

        materials.append(key)

    i=0
    for i in range(len(materials)):
        element_XML = '<data>\n<element_type>' + '</element_type>\n'
        XML = XML + element_XML
        material_XML = '<item>\n<material>' + materials[i] + '</material>\n'
        XML = XML + material_XML
        volume_unit_XML = '<volume_unit>' + global_dict['Material Quantities'][materials[i]][0] +
        '</volume_unit>\n'
        volume_XML = '<volume>' + str(global_dict['Material Quantities'][materials[i]][1]) +
        '</volume>\n'

        XML += volume_unit_XML + volume_XML + '</item>\n'
        XML += '</data>\n'

    i+=1

    XML +=
    '</table>\n<table>\n<dictionary>Program</dictionary>\n<data>\n<element_type></element_type>\n<item
    >\n<program>' + global_dict['Program']['Program'] + '</program>\n</item>\n</data>\n</table>\n'
    XML += '</global>'
    return XML

def filter_data(self):
    """This method return a global dictionary in XML, containing four distinct results obtained by calling
    the following methods: elements_properties; elements_quantities; material_quantities; program. The XML
    structure is built to be used in Microsoft Excel with the function "FILTERXML".
    """

    XML = '<result>\n<header>\n'+ '<information>' + 'Joao Sa; MIEC FEUP. 2013/2014' +
    '</information>\n' + '</header>\n'

    global_dict = { }
    elements_properties = self.elements_properties()
    elements_quantities = self.elements_quantities()
    material_quantities = self.materials_quantities()
    program = self.program()
    global_dict['Elements Properties'] = elements_properties
    global_dict['Elements Quantities'] = elements_quantities
    global_dict['Material Quantities'] = material_quantities
    global_dict['Program'] = program

```

```

number_elements = []
list_of_entities = self.list_of_entities

number_elements.append(str(len(list_of_entities['IFCCOLUMN'])) + ' Columns')
number_elements.append(str(len(list_of_entities['IFCBEAM'])) + ' Beams')
number_elements.append(str(len(list_of_entities['IFCSLAB'])) + ' Slabs')
number_elements.append(str(len(list_of_entities['IFCFOOTING'])) + ' Footings')

```

```

XML += '<body>\n<number>' + str(number_elements)[1:-1] + '</number>\n<properties>Elements
Properties\n'

```

```

elements = []
for key, value in dict.items(global_dict['Elements Properties']):
    elements.append(key)

```

```

i=0
for i in range(len(elements)):

```

```

    for j in range(len(global_dict['Elements Properties'][elements[i]])):

```

```

        if global_dict['Elements Properties'][elements[i]][0] == "Or this type of element doesn't exist, or
it wasn't possible to determine it's properties.":

```

```

            element_XML = '<'+elements[i][:-1].lower()+str(j+1)+'>' + elements[i] + '\n'
            entity_XML = '<ifc_entity>' + 'Error' + '</ifc_entity>\n'
            name_XML = '<element_name>' + 'Error' + '</element_name>\n'
            material_XML = '<material>' + 'Error' + '</material>\n'
            length_unit_XML = '<length_unit>' + 'Error' + '</length_unit>\n'
            dimension_X_XML = '<dimension_X>' + 'Error' + '</dimension_X>\n'
            dimension_Y_XML = '<dimension_Y>' + 'Error' + '</dimension_Y>\n'
            depth_XML = '<depth>' + 'Error' + '</depth>\n'
            area_unit_XML = '<area_unit>' + 'Error' + '</area_unit>\n'
            area_XML = '<area>' + 'Error' + '</area>\n'
            volume_unit_XML = '<volume_unit>' + 'Error' + '</volume_unit>\n'
            volume_XML = '<volume>' + 'Error' + '</volume>\n'

```

```

            XML += element_XML + entity_XML + name_XML + material_XML + length_unit_XML +
dimension_X_XML + dimension_Y_XML + depth_XML + area_unit_XML + area_XML +
volume_unit_XML + volume_XML + '</'+elements[i][:-1].lower()+str(j+1)+'>\n'

```

```

        elif global_dict['Elements Properties'][elements[i]][0] != "Or this type of element doesn't exist, or
it wasn't possible to determine it's properties.":

```

```

            element_XML = '<'+elements[i][:-1].lower()+str(j+1)+'>' + elements[i] + '\n'

```

```

        entity_XML = '<ifc_entity>' + global_dict['Elements Properties'][elements[i]][j][0] +
        '</ifc_entity>\n'
        name_XML = '<element_name>' + global_dict['Elements Properties'][elements[i]][j][1] +
        '</element_name>\n'
        material_XML = '<material>' + global_dict['Elements Properties'][elements[i]][j][2] +
        '</material>\n'
        length_unit_XML = '<length_unit>' + global_dict['Elements Properties'][elements[i]][j][3] +
        '</length_unit>\n'
        dimension_X_XML = '<dimension_X>' + str(global_dict['Elements
        Properties'][elements[i]][j][4]) + '</dimension_X>\n'
        dimension_Y_XML = '<dimension_Y>' + str(global_dict['Elements
        Properties'][elements[i]][j][5]) + '</dimension_Y>\n'
        depth_XML = '<depth>' + str(global_dict['Elements Properties'][elements[i]][j][6]) +
        '</depth>\n'
        area_unit_XML = '<area_unit>' + global_dict['Elements Properties'][elements[i]][j][7] +
        '</area_unit>\n'
        area_XML = '<area>' + str(global_dict['Elements Properties'][elements[i]][j][8]) + '</area>\n'
        volume_unit_XML = '<volume_unit>' + global_dict['Elements Properties'][elements[i]][j][9] +
        '</volume_unit>\n'
        volume_XML = '<volume>' + str(global_dict['Elements Properties'][elements[i]][j][10]) +
        '</volume>\n'

        XML += element_XML + entity_XML + name_XML + material_XML + length_unit_XML +
        dimension_X_XML + dimension_Y_XML + depth_XML + area_unit_XML + area_XML +
        volume_unit_XML + volume_XML + '</'+elements[i][:-1].lower()+str(j+1)+'>\n'
        j+=1
    i+=1
    XML += '</properties>\n<quantities>Quantities by element type\n'

    i=0
    for i in range(len(elements)):

        for j in range(len(global_dict['Elements Quantities'][elements[i]])):
            if global_dict['Elements Quantities'][elements[i]][j][1] == "Or this type of element doesn't exist,
            or it wasn't possible to determine it's properties.":
                element_XML = '<'+elements[i].lower()+>' + elements[i] + '\n'
                material_XML = '<material>' + 'Error' + '</material>\n'
                volume_unit_XML = '<volume_unit>' + 'Error' + '</volume_unit>\n'
                volume_XML = '<volume>' + 'Error' + '</volume>\n'

                XML += element_XML + material_XML + volume_unit_XML + volume_XML +
                '</'+elements[i].lower()+>'

```

```
elif global_dict['Elements Quantities'][elements[i]][j][1] != "Or this type of element doesn't exist,
or it wasn't possible to determine it's properties.":
```

```
    element_XML = '<'+elements[i].lower()+>' + elements[i] + '\n'
```

```
    number_materials_XML = '<number>'+str(len(global_dict['Elements
Quantities'][elements[i]]))+'</number>\n'
```

```
    material_XML = '<material'+str(j+1)+>' + global_dict['Elements Quantities'][elements[i]][j][0]
```

```
    volume_unit_XML = '<volume_unit>' + global_dict['Elements
Quantities'][elements[i]][j][1][0] + '</volume_unit>\n'
```

```
    volume_XML = '<volume>' + str(global_dict['Elements Quantities'][elements[i]][j][1][1]) +
'</volume>\n'
```

```
    XML += element_XML + number_materials_XML + material_XML + volume_unit_XML +
volume_XML + '</material'+str(j+1)+>\n' + '</'+elements[i].lower()+>\n'
```

```
    j+=1
```

```
    i+=1
```

```
XML += '</quantities>\n<global_quantities>Global quantities\n'
```

```
materials = []
```

```
for key, value in dict.items(global_dict['Material Quantities']):
```

```
    materials.append(key)
```

```
XML += '<number>'+ str(len(materials)) + ' Material(s)</number>\n'
```

```
i=0
```

```
for i in range(len(materials)):
```

```
    material_XML = '<material'+str(i+1)+>' + materials[i] + '\n'
```

```
    volume_unit_XML = '<volume_unit>' + global_dict['Material Quantities'][materials[i]][0] +
'</volume_unit>\n'
```

```
    volume_XML = '<volume>' + str(global_dict['Material Quantities'][materials[i]][1]) +
'</volume>\n'
```

```
    XML += material_XML + volume_unit_XML + volume_XML + '</material'+str(i+1)+>\n'
```

```
    i+=1
```

```
XML += '</global_quantities>\n'
```

```
XML += '<program>' + global_dict['Program']['Program'] + '</program>\n'
```

```
XML += '</body>\n</result>'
```

```
return XML
```

A8

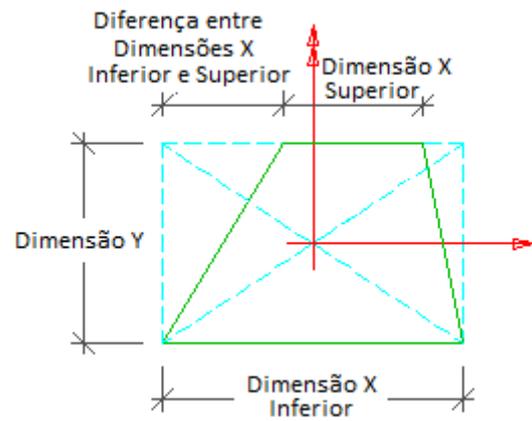
ENTIDADES REPRESENTATIVAS DE DIFERENTES TIPOS DE SEÇÕES NO IFC

Na tabela seguinte, apresenta-se uma lista exhaustiva das entidades que podem caracterizar a seção de um elemento no modelo IFC, indicando-se, também, o tipo de perfil que esta representa e uma representação esquemática tipo. Este estudo incide em seções de geometria regular e não regular, seções ocas ou tubulares e perfis metálicos.

ENTIDADE	TIPO DE PERFIL	REPRESENTAÇÃO ESQUEMÁTICA
Geometria Regular		
IfcRectangleProfileDef	Retangular	
IfcRoundedRectangleProfileDef	Retangular com raios de concordância nos cantos	
IfcEllipseProfileDef	Elítica	

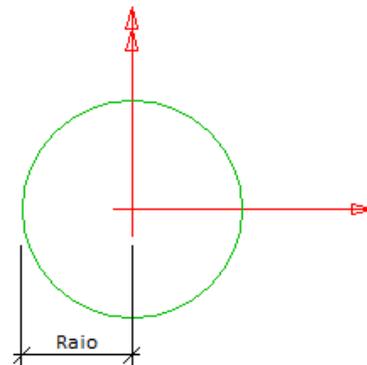
IfcTrapeziumProfileDef

Trapezoidal



IfcCircleProfileDef

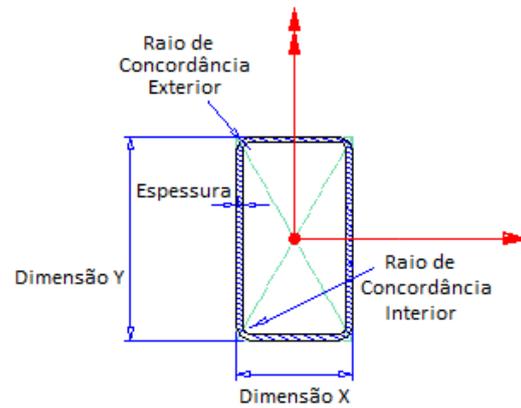
Circular



Seções Tubulares

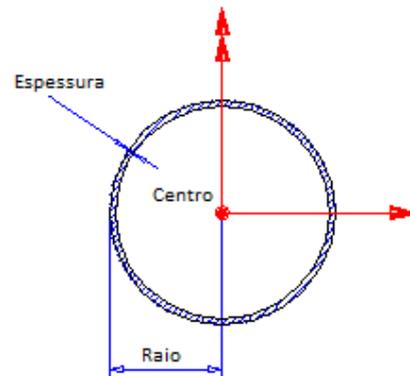
IfcRectangleHollowProfileDef

Retangular Oco, sendo opcional a introdução de raios de concordância



IfcCircleHollowProfileDef

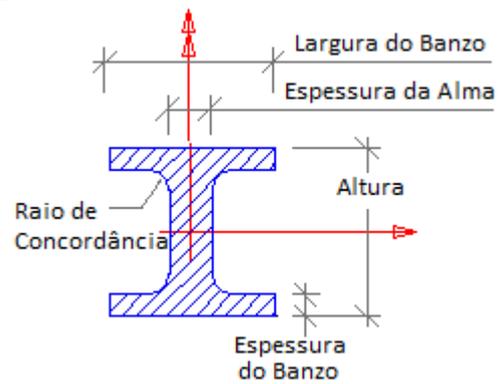
Oco Circular (tubo)



Perfis Metálicos

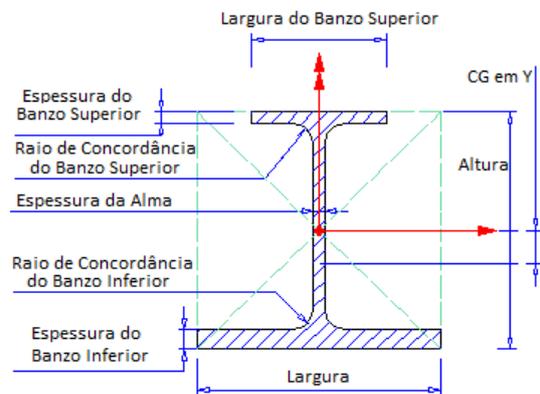
IfcIShapeProfileDef

"I"



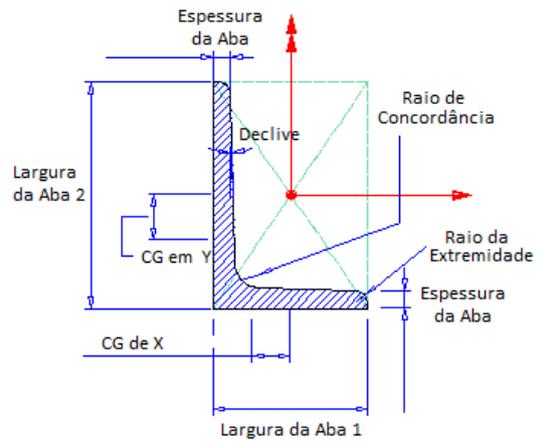
IfcAsymmetricIShapeProfileDef

"I" (assimétrico)



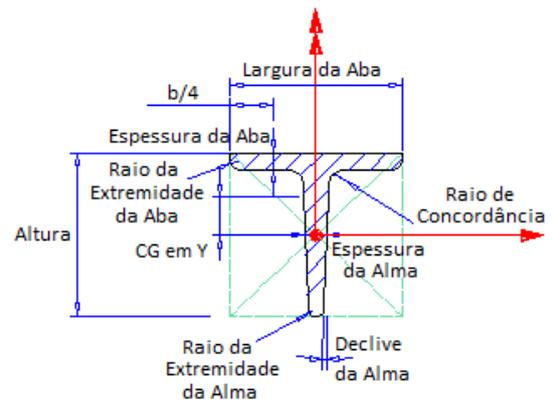
IfcLShapeProfileDef

"L" (cantoneira)



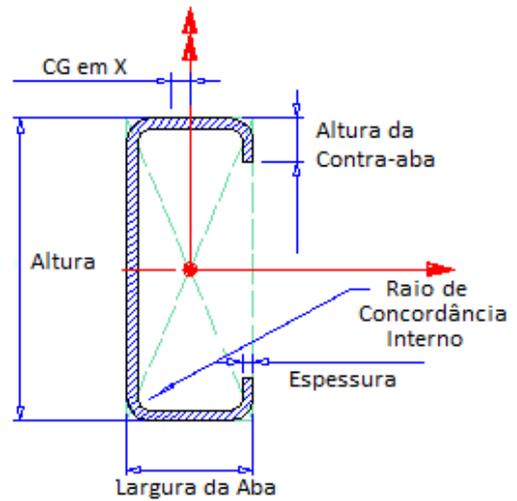
IfcTShapeProfileDef

"T"



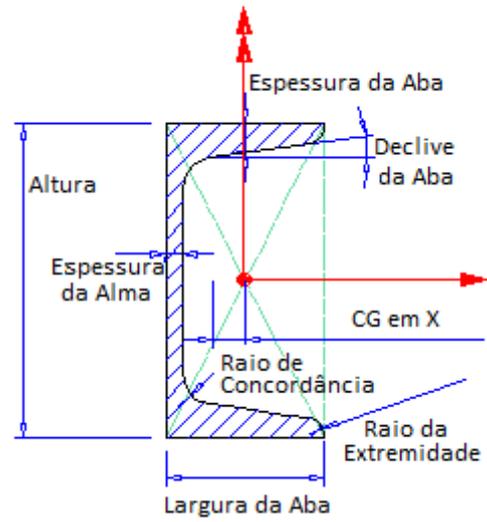
IfcCShapeProfileDef

"C"



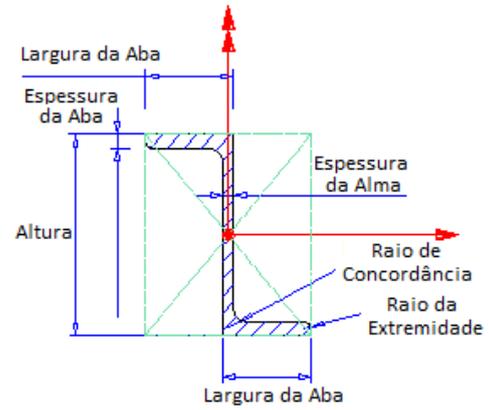
IfcUShapeProfileDef

“U”



IfcZShapeProfileDef

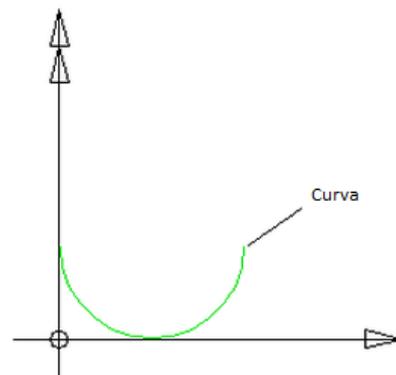
“Z”



Geometria não Regular

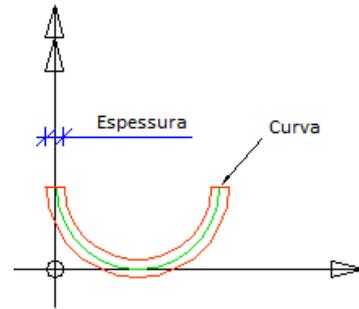
IfcArbitraryOpenProfileDef

Bidimensional arbitrário aberto



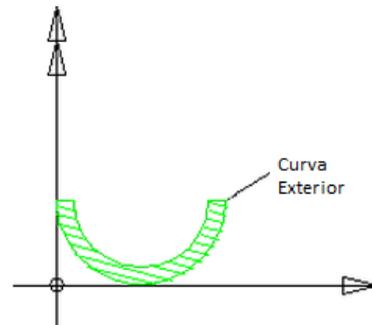
IfcCenterLineProfileDef

Bidimensional
arbitrário aberto,
sem auto
interseção



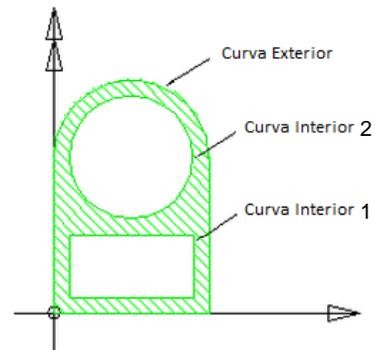
IfcArbitraryClosedProfileDef

Bidimensional
arbitrário fechado



IfcArbitraryProfileDefWithVoids

Bidimensional
arbitrário fechado
perfurado



Posto isto, expõem-se todos os atributos que formam as entidades apresentadas, tanto para o IFC2x3 como para o IFC 4. Sempre que um atributo for representado a negrito ou a rasurado significa que este é adicionado ou eliminado na transição do IFC 2x3 para o IFC 4, respetivamente.

ENTIDADES E RESPECTIVOS ATRIBUTOS	(IFC 2x3 E IFC4)
Seções de Geometria Regular	
#n = IfcRectangleProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Dimensão X", "Dimensão Y")
#n = IfcRoundedRectangleProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Dimensão X", "Dimensão Y", "Raio")
#n = IfcEllipseProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Semi Eixo 1", "SemiEixo 2")
#n = IfcTrapeziumProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Dimensão X Inferior", "Dimensão X Superior", "Dimensão Y", "Diferença entre Dimensões X Inferior e Superior")
#n = IfcCircleProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Raio")
Seções Tubulares	
#n = IfcRectangleHollowProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Dimensão X", "Dimensão Y", "Espessura", "Raio da Concordância Interior", "Raio da Concordância Exterior")
#n = IfcCircleHollowProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Raio", "Espessura")
Perfis Metálicos	
#n = IfcIShapeProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Largura do Banzo", "Altura", "Espessura da Alma", "Espessura do Banzo", "Raio de Concordância", " Raio da Extremidade do Banzo ", " Declive do Banzo ")
#n = IfcAsymmetricIShapeProfileDef ¹	("Tipo de perfil", "Nome de Perfil", "Posição", "Largura", "Altura", "Espessura da Alma", "Espessura do Banzo", "Raio de Concordância", "Largura do Banzo Superior", "Espessura do Banzo Superior", "Raio de Concordância do Banzo Superior", " CG em Y ", " Raio da Extremidade da Aba Inferior ", " Declive da Aba Inferior ", " Raio da Extremidade da Aba Superior ", " Declive da Aba Superior ")
#n = IfcLShapeProfileDef	("Tipo de perfil", "Nome de Perfil", "Posição", "Largura da Aba 2", "Largura da Aba 1", "Espessura da Aba", "Raio de Concordância", "Raio da Extremidade", "Declive", " CG em X ", " CG em Y ")

¹ Os atributos que eram herdados da entidade "IfcIShapeProfileDef" no IFC 2x3 passam a ser definidos diretamente na entidade apresentada para o IFC 4.

```
#n = IfcTShapeProfileDef ("Tipo de perfil", "Nome de Perfil", "Posição",
    "Altura", "Largura da Aba", "Espessura da Alma",
    "Espessura da Aba", "Raio de Concordância", "Raio da
    Extremidade da Aba", "Raio da Extremidade da Alma",
    "Declive da Alma", "Declive da Aba", "CG em Y")

#n = IfcCShapeProfileDef ("Tipo de perfil", "Nome de Perfil", "Posição",
    "Altura", "Largura da Aba", "Espessura", "Altura da
    Contra-aba", "Raio de Concordância Interno", "CG em
    X")

#n = IfcUShapeProfileDef ("Tipo de perfil", "Nome de Perfil", "Posição",
    "Altura", "Largura da Aba", "Espessura da Alma",
    "Espessura da Aba", "Raio de Concordância", "Raio da
    Extremidade", "Declive da Aba", "CG em X")

#n = IfcZShapeProfileDef ("Tipo de perfil", "Nome de Perfil", "Posição",
    "Altura", "Largura da Aba", "Espessura da Alma",
    "Espessura da Aba", "Raio de Concordância", "Raio da
    Extremidade")
```

Seções de Geometria não Regular

```
#n = IfcArbitraryOpenProfileDef ("Tipo de perfil", "Nome de Perfil", "Curva")

#n = IfcCenterLineProfileDef ("Tipo de perfil", "Nome de Perfil", "Curva",
    "Espessura")

#n = IfcArbitraryClosedProfileDef ("Tipo de perfil", "Nome de Perfil", "Curva
    Exterior")

#n = IfcArbitraryProfileDefWithVoids ("Tipo de perfil", "Nome de Perfil",
    "Curva Exterior", "Curvas Interiores")
```