CERVANTES AYRES FILHO

ACESSO AO MODELO INTEGRADO DO EDIFÍCIO

CURITIBA

2009

CERVANTES AYRES FILHO

ACESSO AO MODELO INTEGRADO DO EDIFÍCIO

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós-Graduação em Construção Civil do Setor de Tecnologia da Universidade Federal do Paraná.

Orientador: Professor Dr. **Sergio Scheer**

CURITIBA
2009

Sumário

Lista de Figuras

Lista de Anexos

Resumo

A modelagem de produto na construção, atualmente conhecida pelo termo BIM (*Building Information Modeling*) é uma ferramenta com reconhecido potencial para aumentar significativamente a qualidade dos processos e dos produtos da indústria da construção civil. Ela está modificando gradualmente o modo como a informação flui entre as diferentes fases da produção de edificações e torna-se rapidamente um fator determinante para o posicionamento estratégico das empresas e dos profissionais da indústria da construção. A principal ferramenta da modelagem de produto na construção é o modelo do edifício, um repositório de informações acessado por todos os profissionais envolvidos no desenvolvimento do edifício, da sua concepção à sua construção, manutenção e disposição final. O modelo do edifício representa as características físicas e funcionais dos componentes da edificação, em um ambiente multidimensional onde elas podem ser testadas e aprimoradas antes do início das obras. Diferentes disciplinas da construção utilizam aplicações computacionais próprias que acessam o modelo do edifício, extraem e processam os dados, e produzem informações que são então agregadas ao modelo, refinando-o incrementalmente. Neste trabalho é apresentada uma revisão sobre as origens e conceitos fundamentais da modelagem de produto na construção. Também são apresentados experimentos realizados para demonstrar as várias formas de acessar os dados de modelos de edifícios e as suas potenciais vantagens para o desenvolvimento de aplicações computacionais para a indústria da construção.

Palavras-chave: Modelagem de produto na construção, BIM, Modelo Integrado do Edifício.

Abstract

1
Introdução

"Assessing the impact of technological revolutions – especially fast paced ones like those induced by information technology – is difficult, especially when we are still in the midst, if not at the very beginning of the revolution" – Yehuda Kalay, 2005.

A indústria da construção vem sendo pressionada por mudanças na forma de produção em decorrência de fatores econômicos e sociais ocorridos nas últimas décadas (NASCIMENTO e SANTOS, 2006). É crescente o nível de conscientização da população a respeito dos impactos da sociedade, do mercado e do próprio Estado sobre o meio ambiente, e o amadurecimento da sociedade civil (NOVAES, 1996). O reconhecimento dos direitos do consumidor trouxe a necessidade de processos gerenciais que garantam a qualidade dos produtos e serviços, bem como a sua conformidade com sistemas de normatização nacionais e internacionais (PAULA e MELHADO, 2005). Ao mesmo tempo, a popularização da informática modificou a relação das pessoas com a tecnologia, e transformou consideravelmente tanto o próprio processo de projeto de edifícios quanto as demandas dos clientes (KOWALTOWSKI *et al.*, 2006). A busca por vantagens competitivas torna-se rapidamente assunto de extrema importância, visto que são cada vez mais exigidos, além da óbvia qualidade da solução projetual, menores custos e prazos de entrega, serviços mais flexíveis e soluções diferenciadas ou personalizadas (MELHADO e GRILO, 2003).

Na fase de projetos são definidas as principais diretrizes dos empreendimentos na indústria da construção. Ela tem influência direta nos custos, prazos e métodos de produção (MELHADO, 1994; TZORTZOPOULOS, 1999; MELHADO e AQUINO, 2001; BERTEZINI, 2006). Portanto, um bom processo de projeto, conduzido com o auxílio de ferramentas de informação adequadas, é um pilar fundamental para a qualidade dos processos de construção e dos edifícios resultantes (MOUM, 2006). Em 1982, o *Construction Industry Institute* (CII), dos Estados Unidos, publicou um relatório defendendo os benefícios obtidos pelo investimento na qualidade do projeto, no

sentido de racionalizar a construção civil. No relatório foi estimado que o investimento na melhoria dos projetos pode resultar em uma economia de até vinte vezes o seu valor na fase de execução das obras. Cinco anos após, o CII publicou um guia para implementação dos conceitos de construtibilidade – a capacidade do projeto em fomentar a economia de recursos, segurança e facilidade de uso durante o ciclo de vida do edifício, atendendo perfeitamente às especificações do cliente. Dos quatorze tópicos abordados neste guia, seis pertenciam à fase de concepção do projeto, sete pertenciam à fase de desenvolvimento do projeto e contratação de empreiteiros, e apenas um à fase de execução da obra (LAM *et al.*, 2005). No Brasil, o Programa Setorial de Qualidade (PSQ) definiu a melhoria dos projetos como elemento essencial na obtenção de maior qualidade e processos mais racionalizados na indústria da construção (PSQ, 1997).

Em consequência da sua relevância para a totalidade do processo de construção, a fase de projetos, quando mal conduzida, é responsável por boa parte dos problemas ocorridos nas fases subsequentes. Estudos indicam que erros na documentação da construção são responsáveis pela maior parte das falhas ocorridas nas obras. Na Europa, erros e falta de informação no projeto são causas de 36 a 49% das falhas nas construções, e no Brasil, pesquisa de 1989 indicava o projeto como origem de 46% destas falhas, contra 22% oriundas da execução da obra (MELHADO, 1994). Em pesquisa mais recente, foi demonstrado que projetos inadequados eram responsáveis por 60% dos defeitos patológicos apresentados pelas construções estudadas (AMBROZEWICZ, 2003). Fabrício e outros autores citam entre as principais deficiências dos projetos a ausência de informações necessárias às atividades de produção do edifício, ou mesmo a desconsideração da fase de produção nas soluções adotadas pelo projeto. Isso leva as equipes de obra a decidir por si próprias a respeito das características que não foram especificadas no projeto (FABRICIO *et al.*, 1998). Por estes motivos, atualmente existe entre os pesquisadores a consciência da relação direta entre a melhoria do processo de projeto e o sucesso no estabelecimento de novos critérios de qualidade e competitividade para a indústria da construção, tanto no Brasil como em outros países (ROMANO *et al.*, 2001).

A fase de projetos é essencialmente uma sequência de aprimoramento de um conjunto de informações a ser transmitido para as fases subsequentes. Mesmo projetos típicos na indústria da construção produzem uma enorme quantidade de informação, e por isso os benefícios do uso de tecnologias da informação (TIs) são óbvios. Entretanto, há uma grande distância entre a pesquisa em tecnologia da informação aplicada à construção civil e o os métodos realmente praticados pela indústria no cotidiano, tanto no Brasil como nos outros países. A indústria da construção tem um caráter eminentemente conservador, o que torna difícil a incorporação dos avanços e a mantém tecnologicamente atrasada em relação a outros segmentos da indústria. (NASCIMENTO e SANTOS, 2002).

Mesmo os sistemas de informação já existentes, que já poderiam ser utilizados como apoio ao projeto, são pouco aplicados na prática profissional. Bertezini, estudando empresas do setor, ressalta que os métodos de retro-alimentação e identificação das informações necessárias à tomada de decisões são ineficazes, dificultando a formação de um corpo de conhecimentos. A inexistência de um sistema de informações na fase de projeto resulta na reincidência em falhas já apresentadas em outras situações e no aumento dos custos operacionais (BERTEZINI, 2006).

Ainda não há métodos consolidados para a avaliação do impacto da utilização de sistemas de informação no processo de projeto (NASCIMENTO e SANTOS, 2002; LOVE *et al.*, 2005; PEANSUPAP e WALKER, 2005). Porém há perspectivas promissoras em termos de redução de tempo e aumento da qualidade da documentação produzida pela utilização de modelos tridimensionais integrados e com grande conteúdo semântico no lugar de desenhos bidimensionais desconectados, num processo conhecido como modelagem de produto na construção (RIVARD, 2000; TSE *et al.*, 2005; BIRX, 2006; MIKALDO, 2006; HARTMANN e FISCHER, 2008).

A modelagem de produto na construção pode proporcionar um melhor gerenciamento dos fluxos de informações e das suas transformações durante as atividades realizadas em todo o ciclo de vida do edifício, da concepção à construção, utilização e demolição. A principal ferramenta neste processo é o modelo do edifício, um repositório integrado de informações acessado por todos os envolvidos no

desenvolvimento do edifício. Durante o processo de projetos, idealmente, as diferentes disciplinas da construção utilizariam aplicações especializadas para extrair e processar os dados do modelo, realizando análises e produzindo conhecimentos que são então agregados a ele. A informação acrescentada por cada aplicação seria reutilizada pelos outros envolvidos no projeto, em um processo de refinamento incremental do modelo. Essa abordagem geraria um ambiente de "construção virtual", no qual todas as características do edifício poderiam ser testadas e aprimoradas antes do início das obras. Porém, os métodos para realizar este acesso aos dados dos modelos ainda são pouco documentados e pouco utilizados, limitando o desenvolvimento de um maior número de aplicações para este ambiente cooperativo.

O objetivo do presente trabalho é demonstrar as diferentes formas para acessar os modelos de edifícios e extrair dados que podem então ser processados para a geração de novas informações automaticamente. Primeiramente é apresentada uma revisão das principais características da modelagem de produto na construção, e a sua relação com os métodos de desenvolvimento de projetos atualmente empregados na indústria. Em seguida são relatados os desenvolvimentos e os resultados de uma série de experimentos com as várias formas de acesso aos modelos de edifícios e as ferramentas necessárias para cada uma.

1.1 Estrutura da dissertação

Esta dissertação é dividida em duas partes principais: a revisão das características da modelagem de produto (capítulos 2 e 3) e o desenvolvimento de experimentos de acesso aos dados dos modelos de edifícios (capítulos 4 e 5). O presente capítulo introduz rapidamente o leitor ao contexto e desafios da indústria da construção. Os demais capítulos desenvolvem o tema na seguinte ordem:

- Capítulo 2: são apresentados os princípios da modelagem de produtos e a sua origem na indústria da manufatura.
- Capítulo 3: apresenta as características da aplicação da modelagem de produto na indústria da construção.
- Capítulo 4: descreve os experimentos realizados para demonstrar a viabilidade do acesso aos modelos de edifícios e geração de informações a partir deles.
- Capítulo 5: apresenta as considerações finais sobre os resultados dos experimentos, e sugestões para o desenvolvimento de trabalhos complementares a este.
- Capítulo 6: referências.
- Apêndices: são apresentados os artigos publicados durante o desenvolvimento da dissertação e os algoritmos produzidos durante os experimentos.

2
Modelagem de Produto

"It is only worthwhile to make drawings on the computer if you get something more out of the drawing than just a drawing." — Ivan Sutherland, 1963.

Modelagem é a criação de representações – chamadas modelos – de fenômenos ou sistemas, com o intuito de melhor compreender a sua natureza e prever o seu comportamento. Modelos traduzem para uma forma simplificada um conjunto de entidades complexo demais para ser apreendido em sua totalidade (MAHDAVI, 2003). Essa abstração permite transmitir apenas as características essenciais do sistema representado, protegendo os receptores da informação de detalhes que prejudicariam a sua compreensão (TURK, 2001). Na indústria da construção são utilizadas modelagens em várias etapas do projeto de edifícios, da elaboração de esquemas explicando conceitos e equações prevendo comportamentos físicos à criação de protótipos para demonstrar a factibilidade das idéias. O próprio processo de projeto como um todo pode ser considerado uma modelagem: um refinamento sucessivo de um modelo conceitual, que é o edifício proposto (TAKEDA *et al.*, 1990).

Alguns tipos de modelagem utilizam representações tridimensionais digitais de um sistema, para fornecer uma visualização realista dos seus elementos e dos seus comportamentos (BEUCKE *et al.*, 2005). Por exemplo, na indústria da construção brasileira – notadamente na fase de projetos – o termo modelagem está fortemente associado à criação de representações computacionais tridimensionais das edificações, também chamadas maquetes eletrônicas (SPERLING, 2002). Considerando a definição etimológica, qualquer representação de um edifício, independente do conteúdo e da ferramenta utilizada, pode ser chamada um modelo: esboços, estudos de volumetria, desenhos técnicos em papel ou digitais, maquetes físicas ou eletrônicas, detalhamentos, etc. Neste trabalho, o termo modelo refere-se ao resultado de uma abordagem de desenvolvimento de produtos industriais, chamada modelagem de produto. Um modelo de produto pode fornecer vários tipos de representação, entre

elas as citadas anteriormente, com a vantagem de integrá-las em um único repositório, garantindo a consistência dos dados.

O desenvolvimento de produtos industriais é composto por uma série de operações complexas que podem ser organizadas em dois grandes grupos: o processamento de informações e o processamento de materiais. Dos primeiros esboços até o produto concluído, informação de vários tipos é gerada, transformada e transmitida entre as diversas fases do desenvolvimento, ao mesmo tempo em que ocorrem várias transformações nos materiais. Estes dois grupos de operações são fortemente relacionados e idealmente devem ser sincronizados. A principal maneira de se obter essa sincronização é representar toda a informação fundamental sobre o produto digitalmente, a partir das primeiras fases do desenvolvimento, na forma de um modelo do produto, que é então utilizado para coordenar as atividades da produção. Essa abordagem é denominada modelagem de produto (HOSAKA e KIMURA, 1990). Um modelo de um produto, portanto, é um repositório único de dados sobre este produto, que orienta as atividades desenvolvidas durante todo o seu ciclo de vida (KRAUSE *et al.*, 1993).

2.1 A informática e o desenvolvimento de produtos

A maneira como os produtos eram fabricados sofreu poucas alterações até o final do século XIX, quando os avanços tecnológicos aumentaram os requisitos a serem atendidos pela fase de concepção, e a separação do trabalho por funções fez surgir a necessidade da padronização da comunicação entre as etapas do seu projeto. A formalização desse tipo de comunicação se deu através do uso de desenhos técnicos, e foi concluída na década de 1930 (KRAUSE *et al.*, 1993).

Logo após o fim da segunda grande guerra, o computador foi introduzido como ferramenta de apoio no desenvolvimento de produtos industriais. Desde a sua primeira aplicação, a intenção era integrar os diversos processos envolvidos na fabricação – da concepção ao produto acabado – transmitindo informações entre as etapas do desenvolvimento. No começo da década de 1950, o *Servomechanisms Laboratory*, do *Massachusetts Institute of Technology* (MIT), desenvolveu a primeira fresadora de três eixos controlada automaticamente. A informação que controlava a

máquina era introduzida na forma de fitas de papel perfuradas, preparadas manualmente por um operador que traduzia o desenho técnico detalhado para a forma numérica, e então para os padrões apropriados de furos nas fitas. Não demorou para que o computador fosse envolvido nesse tedioso processo, e no final da mesma década foi desenvolvido um sistema para preparar automaticamente essas fitas de controle, a partir dos desenhos técnicos, ainda feitos em papel (GALLAGER e MITTER, 1990).

2.1.1 O princípio do CAD

Em 1959 foi realizada uma reunião entre membros do *Computer Applications Group* e do Departamento de Engenharia Mecânica, ambos do MIT, na qual foi discutido o uso do computador de um modo mais direto no desenvolvimento de produtos (COONS, 1963). O sistema esboçado nessa reunião seria desenvolvido posteriormente no projeto de pesquisa chamado *Computer-Aided Design* (ROSS, 1961), que acabou emprestando o nome tanto para o novo ramo da tecnologia de desenvolvimento de produtos como para as aplicações computacionais criadas para ele. Algumas das funcionalidades definidas para este sistema eram a descrição de abstrações (esquemas conceituais e idéias), análises físicas e matemáticas, conexão com catálogos (normas técnicas, peças e materiais), interconexão de vários projetistas trabalhando simultaneamente (com atualizações e propagações automáticas da informação), reutilização da informação do desenho, simulações do funcionamento do dispositivo projetado, e até a consideração de diferentes visões sobre a informação, com a organização da representação dos dados em função das muitas disciplinas envolvidas no projeto.

Um dos resultados mais notórios deste projeto de pesquisa foi o sistema Sketchpad, criado por Ivan Sutherland em sua tese de doutoramento no MIT (SUTHERLAND, 1963). Até então, linguagens textuais eram utilizadas para criar representações gráficas no computador, o que Sutherland considerava inadequado e confuso. No sistema Sketchpad, a entrada de dados era realizada por uma interface gráfica, auxiliada por um dispositivo de interface humana chamado *light-pen*. Posteriormente, foi prevista uma versão que incluiria a representação tridimensional,

considerada essencial para o projeto de pesquisa CAD (JOHNSON, 1963). Coons relata uma rápida experiência com o Sketchpad, na qual o sistema o auxilia a resolver cinco problemas de engenharia no decurso de poucas horas (COONS, 1963). Na sua descrição das funcionalidades que o permitiram resolver os problemas rápida e intuitivamente, é perceptível o delineamento de características que ainda são vitais para a modelagem no desenvolvimento de produtos industriais. Sutherland, complementarmente, afirma que a criação de projetos no Sketchpad é uma sequência de etapas orientadas pela funcionalidade dos elementos representados (SUTHERLAND, 1963):

> *"Construction of a drawing with Sketchpad is itself a model of the design process. The locations of the points and lines of the drawing model the variables of a design, and the geometric constraints applied to the points and lines of the drawing model the design constraints which limit the values of design variables. The ability of Sketchpad to satisfy the geometric constraints applied to the parts of a drawing models the ability of a good designer to satisfy all the design conditions imposed by the limitations of his materials, cost, etc. In fact, since designers in many fields produce nothing themselves but a drawing of a part, design conditions may well be thought of as applying to the drawing of a part rather than to the part itself. When such design conditions are added to Sketchpad's vocabulary of constraints, the computer will be able to assist a user not only in arriving at a nice looking drawing, but also in arriving at a sound design."*

Ou seja, em princípio pensava-se em CAD como uma ferramenta de desenvolvimento de produtos com rotinas capazes de auxiliar a concepção, produzir representações, executar análises, prever comportamentos e gerar instruções para a fase de produção, em um ambiente integrado de projeto. Todas essas funcionalidades exigiam um modo de entrada de dados ágil e intuitivo, e a linguagem dos desenhos técnicos, já bastante desenvolvida e codificada, foi adotada para essa finalidade. A mudança do suporte – ou mídia – dos desenhos técnicos do papel para o computador não era um objetivo em si, era apenas uma consequência do uso da ferramenta CAD.

Embora coerente, essa abordagem integrada de desenvolvimento de produtos teve pouca influência sobre os métodos de fabricação por vários anos. A academia continuou a aperfeiçoar definições e protótipos para sistemas integrados de desenvolvimento de produtos, mas a nascente indústria de *softwares* passou a se concentrar no aspecto que podia ser mais facilmente resolvido – a criação de desenhos

no computador, por meio de primitivos geométricos (pontos, linhas, arcos). As aplicações comercialmente disponíveis que ofereciam esta funcionalidade logo passaram a ser conhecidas por CAD, mas em relação aos princípios definidos pelo projeto de pesquisa de Ross, o seu nome poderia ser melhor traduzido por *Computer Aided* ***Drafting*** do que por ***Design***, já que a sua influência sobre o desenvolvimento de produtos era basicamente a mesma das pranchetas de desenho que se esperava que ele substituísse.

Parte do que conduziu a indústria de *softwares* para esta direção foi consequência do baixo poder de processamento dos primeiros computadores. Outro motivo foi a enorme complexidade das atividades envolvidas no projeto de produtos, entre elas a altamente cognitiva e subjetiva fase de concepção. Douglas Ross, líder do grupo de pesquisa CAD, e Jorge Rodriguez afirmaram que um sistema CAD para uso geral deveria possuir uma poderosa organização, pois mesmo os mais simples problemas de projeto envolvem o exercício de muitas disciplinas diferentes e a consideração de muitas atividades relacionadas. Também deveria ser um sistema aberto à expansões e modificações, pois não seria possível prever e programar todas as abordagens imagináveis para solucionar um determinado problema de projeto, e ter a capacidade de manipular e armazenar informações oriundas das mais diversas fontes e em diferentes formatos. Mesmo atualmente não há uma teoria universalmente aceita para a compreensão dos processos de concepção durante o projeto (KOWALTOWSKI et al., 2006).

Além de aplicar uma poderosa inteligência artificial para resolver esses problemas, o sistema CAD ideal deveria disponibilizar ao operador a capacidade de desenvolver as suas habilidades de projeto de modo que lhe fosse completamente natural, sem que ele fosse consciente que as suas ações de projeto desencadeiam um grande número de operações computacionais altamente complexas (ROSS e RODRIGUEZ, 1963).

Anderl e Grabowski, duas décadas depois dos primeiros trabalhos científicos sobre o CAD, afirmaram que a utilização de computadores na produção industrial deveria ser precedida por uma criteriosa análise de todas as atividades envolvidas,

para que fossem definidas as trocas de informações entre elas. Porém, observaram que a maioria dos sistemas CAD ofereciam apenas a possibilidade de descrever peças individualmente, do mesmo modo que era feito com desenhos técnicos em papel. Os dados sobre a tecnologia aplicada, como acabamento de superfícies e tolerâncias dimensionais, necessários para o planejamento da produção, eram introduzidos posteriormente, com o auxílio de diálogos de entrada ou linguagens de programação (GRABOWSKI e ANDERL, 1983).

Já na década de 1990, Hank Pels afirmou que mesmo nas situações em que representações mais ricas do que os desenhos feitos em computador eram empregadas, havia o problema da transferência da informação. Desde a introdução dos primeiros sistemas CAD, modelos de produtos vinham sendo digitalizados, porém com pouco compartilhamento de dados entre diferentes disciplinas e atividades envolvidas no processo de fabricação. Os modelos, segundo ele, não eram transmitidos entre computadores, e sim manuseados por operadores, traduzidos e transformados em outros modelos, da mesma maneira que era praticada com os desenhos técnicos em papel, mas com a desvantagem do aumento da complexidade da mídia. Uma das possíveis causas para isso era o fato de se considerar os dados necessários para o projeto do produto razoavelmente estáveis ou até mesmo estáticos, em comparação com os dados financeiros e logísticos, por exemplo. Por isso, ainda fazia sentido distribuí-los em papel, na forma de tabelas e manuais técnicos, e um modelo de produto que integrasse essa informação não era considerado necessário (PELS, 1996).

2.1.2 O princípio da modelagem de produto

Apesar da idéia de integração do processo de produção ter sido a motivação dos primeiros sistemas CAD, foram as mudanças econômicas ocorridas a partir do final da década de 1970 que deram novo impulso ao tema e influenciaram os trabalhos que propuseram as atuais definições de modelagem de produto. A globalização dos mercados e a necessidade de fabricar produtos com maior qualidade, menor custo de produção e em menos tempo, deram origem a novas estratégias de desenvolvimento de produtos. Entre elas, a interconexão dos vários aspectos técnicos e gerenciais

envolvidos na produção; a produção enxuta; a engenharia simultânea; e o conceito de ciclo de vida do produto, que estendeu a responsabilidade do projetista para o campo dos impactos ambientais e a saúde dos usuários.

Estas novas estratégias de desenvolvimento tinham focos e abordagens distintos, porém compartilhavam uma necessidade fundamental por tecnologias de informação avançadas, que permitissem integrar e coordenar as diferentes visões sobre o produto durante o projeto, a fabricação e a operação (KRAUSE *et al.*, 1993). Sistemas computacionais já haviam demonstrado o seu potencial na racionalização de várias etapas isoladas do desenvolvimento de produtos, e surgiu a tendência para a integração dos diferentes sistemas através do fluxo digital de informações, em substituição à manipulação de diferentes modelos. Este fluxo teria o potencial para reduzir os custos de produção através da eliminação de atividades de re-entrada de dados, da redução de erros, atividades de controle e procedimentos de teste, e da disponibilização rápida e completa de informações sobre o produto e a sua produção, o que aumentaria a velocidade do processamento de pedidos e a qualidade do produto (GRABOWSKI e ANDERL, 1983).

A ferramenta principal nesse fluxo de informações mencionado por Grabowski e Anderl era a modelagem do produto. Em um retorno aos princípios originais do CAD, ele observou que a adoção da modelagem permitiria que os projetistas concebessem e validassem os novos constructos, e posteriormente comunicassem corretamente os detalhes de construção às fases de produção. Durante a concepção, o modelo do produto auxilia os usuários a explorar, documentar, compreender e predizer certas propriedades e comportamentos dos elementos representados. A validação é a verificação da conformidade do constructo proposto contra os diversos requisitos estabelecidos para a sua fabricação e operação, e a sua importância cresce conforme aumenta a complexidade dos produtos. O modelo do produto permite que sejam utilizadas listas de verificação a simulações computacionais para auxiliar o projetista nesta fase. Ele também proporciona uma comunicação mais eficiente entre as fases do desenvolvimento do produto, uma característica necessária para processos industriais que tornam-se cada vez mais complexos e envolvem um número crescente de pessoas e organizações, muitas vezes distantes geograficamente (PELS, 1996; MAHDAVI, 2003).

Por proporcionar a integração dos diversos sistemas utilizados nos processos de desenvolvimento, a modelagem de produto foi considerada uma tecnologia chave no aumento da produtividade e para a sobrevivência competitiva das companhias (KIMURA *et al.*, 1984).

2.2 Evolução dos modelos de produto

A funcionalidade oferecida pelos modelos de produto evoluiu em conjunto com as possibilidades técnicas dos computadores, as novas concepções de estruturação da informação trazidas pelo estudo dos modelos de dados, e as crescentes demandas por integração dos sistemas de produção. Essas demandas, porém, não foram homogêneas, seja considerando os processos produtivos ou os diferentes segmentos da indústria. A classificação dos tipos de modelos por idade coincide com o aumento da sua complexidade e é útil para ilustrar a origem das funcionalidades dos modelos mais recentes, mas diz respeito apenas ao aspecto técnico, e não à adoção efetiva pela indústria. Mesmo hoje, os primeiros e mais simples tipos de modelos ainda são utilizados como principal meio de transmissão de informações entre processos produtivos em determinados setores, como na construção civil por exemplo.

2.2.1 Modelo geométrico

Os primeiros modelos tinham por objetivo principal representar geometria dos elementos que compunham os produtos. Embora a idéia de representar informações complementares à geometria já estivesse presente desde o início do CAD, a grande dificuldade para equacionar processos complexos utilizando computadores ainda muito simples dirigiu grande parte das pesquisas para a definição de teorias matemáticas que permitissem representar a geometria digitalmente. Eastman e Henrion classificam as primeiras experiências com modelagem no computador em três estágios: a abordagem inicial, denominada *image modeling*, enfocava a produção de desenhos, e sua preocupação maior era representar corretamente objetos tridimensionais em um plano (a tela do monitor) utilizando apenas linhas. O modelo resultante fornecia a informação necessária apenas para a visualização, possivelmente incluindo diferentes perspectivas e eliminação de linhas escondidas (*hidden line*). A segunda abordagem, chamada *geometric modeling*, tinha por objetivo a representação

das superfícies dos sólidos (ou poliedros), para que fosse possível discretizar o seu volume interno e identificar conflitos espaciais. A terceira abordagem seria uma evolução do modelo geométrico, passando a incluir atributos adicionais ao formato, como material, densidade, função, cor ou quaisquer outras informações que fossem relevantes para a produção industrial (EASTMAN e HENRION, 1979).

As bases da modelagem geométrica foram lançadas pelo trabalho de Requicha e Voelcker sobre a geometria construtiva de sólidos (*constructive solid geometry* – CSG), que propunha modelos constituídos por estruturas topológicas e estruturas geométricas (HOFFMANN e JOAN-ARINYO, 2002). Kalay propôs a utilização de bancos de dados relacionais para a representação do modelo CSG como um conjunto de dados relacionados em três níveis hierárquicos: topologia, geometria e transformações espaciais. A topologia é um índice que organiza as relações entre os elementos (primitivos geométricos) que constituem um sólido. A geometria é o nível responsável pela descrição dos primitivos geométricos através de equações matemáticas que possam ser armazenadas digitalmente. O nível das transformações espaciais representa as instâncias dos objetos, criadas a partir de matrizes de transformações, como a rotação ou o redimensionamento do sólido (KALAY, 1985a). Hosaka e Kimura fazem categorização semelhante alguns anos depois (HOSAKA e KIMURA, 1990).

2.2.2 Modelo variacional

O modelo variacional foi introduzido por Voelcker *et al*., no final da década de 1970, e estendeu o modelo geométrico adicionando a possibilidade de definição dos sólidos através da associação de formas básicas definidas por uma linguagem procedimental chamada PADL – *Part and Assembly Description Language* (VOELCKER *et al.*, 1978). A PADL era baseada em funções para as quais eram passados os parâmetros desejados para os elementos geométricos, que então eram construídos durante a execução do programa. Estes parâmetros podiam ser modificados durante a execução, ao contrário da maioria das linguagens de definição de gráficos produzidas até então, que permitiam definir os valores apenas durante a programação do código – ou seja, através de comandos e não de funções. Várias linguagens de definição de sólidos desenvolvidas durante os anos seguintes seguiam o paradigma estabelecido

por Voelcker e seus colegas, como a Glide, por exemplo (EASTMAN e HENRION, 1977). Outro exemplo é a GDL, (*Geometrical Description Language*), que ainda é um dos métodos para definição de sólidos no *software* ArchiCAD (GRAPHISOFT, 2008b).

Uma desvantagem do modelo variacional é a dependência em relação aos *scripts* – os arquivos de texto contendo as funções que geravam a representação dos sólidos durante a execução do programa. Embora pudessem ser bastante complexos e flexíveis, projetar por programação é sempre menos desejável do que fornecer ao projetista ferramentas visuais e um ambiente intuitivo para a construção dos elementos do projeto (HOFFMANN e JOAN-ARINYO, 2002).

2.2.3 Modelo baseado em restrições

O modelo baseado em restrições é outra variação do modelo geométrico, com a introdução da possibilidade de geração de instâncias de sólidos a partir de um conjunto de relações entre as entidades, que precisavam ser mantidas. Quando as entidades são relacionadas, o programa utilizado na modelagem impede operações cujo resultado não atenda as condições de restrição. Os tipos de restrições foram classificados em grupos: geométricas ou dimensionais, equacionais, semânticas e topológicas (HOFFMANN e JOAN-ARINYO, 2002).

Restrições geométricas ou dimensionais mantém entidades relacionadas por uma condição de concentricidade, perpendicularidade, ângulo, distância, etc. As restrições equacionais mantém entidades relacionadas através da avaliação de um atributo calculado a partir da sua geometria ou de variáveis tecnológicas como torque, densidade ou resistência. Restrições semânticas mantém o relacionamento entre as entidades apenas se uma determinada condição for atendida. Restrições topológicas avaliam condições de incidência, conectividade entre elementos, conjuntos ou partes.

2.2.4 Modelo paramétrico

O modelo paramétrico estende o modelo geométrico para além da soma das representações topológicas e geométricas. Eles contém também a metaestrutura a partir da qual novas instâncias dos sólidos podem ser derivadas. Deste modo, é mais apropriado pensar em modelos paramétricos como um conjunto de sólidos

paramétricos, que são classes de sólidos específicos. Uma classe "prisma", por exemplo, contém a informação necessária (a metaestrutura) para criar prismas de qualquer dimensão. A geração de uma instância de objeto é realizada por um algoritmo determinístico, que considera diferentes restrições e avalia parâmetros definidos na estrutura de informação que compõe o sólido paramétrico. Antes do desenvolvimento dos sólidos paramétricos, a modelagem gerava apenas sólidos específicos que, uma vez criados, não podiam ser modificados facilmente. A modelagem paramétrica, ao contrário, não enfoca a representação final do sólido, e sim as etapas envolvidas na sua construção, parametrizando-as e possibilitando que o usuário determine vários resultados para o mesmo objeto a partir da combinação dos seus atributos. A flexibilidade resultante pode ser explorada de muitas maneiras, e se constitui em importante avanço para a aplicação no desenvolvimento de produtos (HOFFMANN e JOAN-ARINYO, 2002). Os sólidos paramétricos são intimamente relacionados à tecnologia de orientação por objetos, de onde surge a denominação objeto paramétrico. Um objeto paramétrico pode ser entendido como uma unidade de informação (ou classe) que encapsula os dados (os parâmetros) e métodos para processá-los (os *scripts*), resultando em uma instância do objeto. Outra analogia para o modelo paramétrico é considerá-lo uma associação das qualidades do modelo geométrico com as do modelo variacional.

2.2.5 Modelo baseado em características (*features*)

O modelo baseado em características é uma especialização do modelo paramétrico. A estrutura de classes proporcionada pela orientação por objetos permite explorar novas classificações hierárquicas além da geometria. Hvam defende que a modelagem de produto é diretamente relacionada a dois conceitos: a engenharia simultânea e a modelagem de características. A engenharia simultânea integra as atividades a partir do planejamento das fases do ciclo de vida do produto, reduzindo o tempo de entrega e os custos de produção. A modelagem de características é um meio de representar as diferentes visões que as várias disciplinas envolvidas no desenvolvimento de um produto têm sobre ele durante o seu ciclo de vida: concepção, topologia, desempenho, tolerâncias, produção, montagem, logística, etc. O modelo baseado em características armazena o conhecimento técnico que é

adicionado ao modelo do produto em cada fase do ciclo de vida pelas várias disciplinas envolvidas, na forma de descrições gerais e de instruções específicas para geração de instâncias do modelo (desenhos técnicos, relatórios quantitativos, sequências de operações). Enquanto a descrição geral é comum para todas as etapas do desenvolvimento do produto, as instâncias dependem de condições específicas de cada fase do ciclo de vida (HVAM, 2001).

Hoffman e Joan-Arinyo acrescentam que o uso de características passou a ser componente padrão da modelagem paramétrica, de onde pode se concluir a atual preferência por denominar de paramétricos modelos que possuem estruturas para descrever também as características do produto. Para os autores, as características proporcionam um vocabulário de alto nível para especificação de operações de criação de formas através da geometria parametrizada, atributos e restrições geométricas. Durante o projeto, as características capturam atributos técnicos explícitos e relacionamentos que auxiliam na definição de produtos, provendo informações essenciais para várias atividades e análises de desempenho. Na fabricação, as características podem ser combinadas para facilitar o planejamento. Para serem úteis, as características devem incorporar três diferentes conceitos: o aspecto geral, o comportamento e o significado técnico. O aspecto geral é a representação geométrica do objeto, definida por fronteiras, árvores de operações booleanas (*CSG tree*) ou por procedimentos criados com linguagens de definição geométrica. O comportamento e o significado técnico são definidos através de atributos e regras, que condicionam o objeto em relação a um contexto específico. Grandes quantidades de especificações poderiam interferir em processos mais ágeis, como a concepção de novos produtos. Por isso é proposto pelos autores que se desenvolvam ferramentas para o reconhecimento automático de características. Nesse processo um modelo geométrico previamente construído seria analisado por algoritmos que identificassem características automaticamente, a partir de um conjunto de regras ou padrões semânticos. Esta abordagem tem recebido crescente atenção por parte da comunidade científica, mas vários desafios ainda precisam ser endereçados e resolvidos para que se obtenha algoritmos confiáveis. Um dos principais desafios é o reconhecimento de características sobrepostas: quando as visões de várias disciplinas

diferentes são combinadas, a topologia resultante pode mudar consideravelmente, e os algoritmos podem ter dificuldade para localizar as características adequadas. Outro desafio é registrar o conhecimento técnico adicionado durante as fases do desenvolvimento do produto: armazenar as diferentes visões sobre os dados do produto que cada disciplina utiliza é tão importante quando armazenar os dados em si. Ainda mais, todas as modificações realizadas sobre uma determinada visão dos dados devem ser propagadas para as outras visões, atualizando todos os profissionais envolvidos (HOFFMANN e JOAN-ARINYO, 2002).

2.3 Modelos de dados de produtos

Modelos de produtos são gerados a partir de arcabouços formados por constructos lógicos que definem a forma e o significado para os dados que representarão o ciclo de vida de um produto (LACROIX e PIROTTE, 1981). Esses arcabouços são criados em um processo de metamodelagem chamado modelagem de dados do produto. Yang e outros autores situam a modelagem de dados do produto na fase imediatamente posterior à análise da estrutura do produto a ser fabricado e do contexto onde será realizada a produção. Depois de criado, o modelo de dados do produto é implementado por vários programas de computador, dando origem ao modelo do produto, que pode ser armazenado em bancos de dados ou em arquivos de formato neutro (YANG *et al.*, 2008). Modelos de produtos são, portanto, sempre instâncias de algum modelo de dados criado previamente.

Antes do advento da modelagem de dados, cada programa definia a sua própria forma de armazenar e recuperar dados, o que envolvia um enorme trabalho de programação e dificultava a adaptação e a evolução dos sistemas, como pode ser percebido no protótipo de sistema CAD integrado criado por Charles Robinson (ROBINSON, 1966). Haynie relata que nos primeiros sistemas de desenvolvimento de projetos era comum que os dados do produto em desenvolvimento fossem indissociáveis dos procedimentos da aplicação CAD. Uma solução melhor, segundo o autor, seria tornar a informação independente da aplicação computacional, o que permitiria a definição de várias visualizações sobre um mesmo dado, por aplicações

diferentes. Para isso, uma nova ferramenta seria adotada para armazenar os dados do projeto – o banco de dados (HAYNIE, 1983).

2.3.1 Bancos de dados

Bancos de dados não só proporcionaram uma forma muito mais eficiente de armazenar e recuperar dados, como também foram a primeira tecnologia que permitiu a modelagem de dados independente do programa que iria processá-los. Em 1975, o *American National Standards Institute* (ANSI) definiu a abordagem básica para a construção de bancos de dados chamada ANSI-SPARC, que ficou mais conhecida como "arquitetura de três esquemas" (*three schema architecture*). Os três esquemas do ANSI-SPARC eram o conceitual – que especifica a estrutura e o significado dos dados e segue as determinações dos processos do negócio; o esquema externo – que especifica a forma de apresentação dos dados e é utilizado pelas aplicações e pelos usuários da informação; e o esquema interno – que especifica a estrutura física do banco de dados, e segue determinações impostas pelo hardware e sistemas operacionais utilizados (PELS, 1996). Eastman observa que no início do uso da tecnologia, ainda havia um certo desentendimento em relação ao conceito de bancos de dados, e o define como uma estrutura de informação (1980a):

> *"Computers today can easily store large amounts of information. It has become a euphemism that any program incorporating more than a hundred or so words of data can be considered to have a database. While this is true in the colloquial sense, the term 'database' has a technical meaning that is related to a set of methods used for managing large amounts of information. Thus a large amount of data, but not using these techniques, is not properly a database, while a small amount of data that does use them legitimately can be still called a database. Thus the term database applies more to the structure of information rather than its size."*

Após a introdução dos bancos de dados, grande parte das instruções para a interpretação da informação passou a residir no modelo de dados (o esquema de organização do banco de dados) e não mais nos programas que iriam acessá-la. Essa liberdade era essencial para o desenvolvimento de estruturas genéricas capazes de armazenar vários tipos de produtos, ou várias instâncias de um tipo de produto, mantendo o significado da informação e permitindo o acesso por diferentes aplicações durante as fases de desenvolvimento do produto. O potencial da aplicação de bancos

de dados na atividade de projeto de produtos, cuja natureza implica justamente na transmissão de informações entre aplicações e disciplinas diferentes, foi extensivamente estudado a partir da década de 1980 (EASTMAN, 1980a; BENNETT, 1982; KALAY, 1985a; EASTMAN e KUTAY, 1989).

Os primeiros sistemas de gerenciamento de bancos de dados, entretanto, destinavam-se a ambientes de negócios, e não contemplavam a complexidade das atividades de projeto, tornando o acesso à informação muito complicado. Bandurski e Jefferson afirmam que a figura do administrador de bancos de dados seria essencial à tecnologia CAD: ele seria um profissional familiarizado com todos os processos do desenvolvimento do produto e com as estruturas de dados necessárias para cada um, capaz de fornecer interfaces que permitissem que o usuário visualizasse os dados do modo mais natural possível, protegido de detalhes desnecessários das estruturas empregadas no banco de dados (BANDURSKI e JEFFERSON, 1975). Para Lacroix e Pirotte, por outro lado, as tecnologias utilizadas pelos primeiros bancos de dados não teriam capacidade suficiente para representar adequadamente a complexa informação de projetos de desenvolvimento de produtos. Tampouco a integração de processos produtivos poderia ser alcançada enquanto a interpretação das estruturas de dados fosse exclusividade de um grupo seleto de especialistas, como os administradores de bancos de dados. Eles observaram que a prática da descrição de modelos de dados através de convenções mal formuladas, que não eram suficientes para esclarecer a natureza da informação a ser armazenada e distribuída, dificultava o compartilhamento do modelo entre especialistas de diferentes disciplinas:

> *"[...] to avoid the 'a little information in data structure diagrams and a lot in accompanying unstructured explanations' syndrome which in practice is typical of many applications expressed in the hierarchic and network data models"*

Para as aplicações CAD, os autores propõem a utilização de um modelo semântico de dados, que introduz a distinção entre os constructos que compõem as estruturas de dados e o valor que eles podem vir a receber (LACROIX e PIROTTE, 1981).

2.3.2 Semântica

Os constructos de um modelo semântico de dados são as entidades, descritas através de elementos atômicos (como classes e propriedades) e os relacionamentos entre estas entidades. Para definir as entidades e os seus relacionamentos, foi necessário criar um novo sistema para notação do modelo de dados – a linguagem de definição de dados. As principais características dessa linguagem são fornecer diferentes abordagens para a descrição dos constructos (relacionamentos, hierarquias, associações), permitir que sejam modelados constructos com riqueza e precisão, e finalmente adequar-se às várias situações e tipos de objetos que possam demandar modelagem, permitindo que os especialistas possam realizar essas atividade de modo que lhes pareça natural. Lacroix e Pirotte desenvolveram uma linguagem de definição de dados para a criação de modelos semânticos de placas de circuito elétrico, chamada ADDL (*A Data Definition Language*). Os modelos semânticos gerados pela ADDL não apenas eram mais facilmente interpretados pelos usuários como garantiam a coerência pelo uso de restrições aplicadas às entidades (LACROIX e PIROTTE, 1981). A ADDL adiantou alguns conceitos que seriam aplicados posteriormente na criação de outras linguagens de definição de dados, como as regras de estruturação de domínio – que definiam os valores possíveis para os tipos complexos de dados (formados por conjuntos de tipos simples), as regras de restrição de domínio, que atuavam em conjunto com as primeiras para determinar qual combinação de domínios poderia constituir um tipo complexo válido, e finalmente a denotação de "objeto" aos tipos complexos que representavam unidades de informação razoavelmente autônomas no processo de projeto.

Mark Haynie relacionou algumas das especificidades às quais os bancos de dados deveriam adequar-se para serem úteis nas aplicações de engenharia: mesmo as unidades de informação mais simples utilizadas na engenharia costumam ser representadas por tipos complexos (um ponto no espaço, por exemplo, é composto por no mínimo três números reais), o acesso em baixo nível precisa ser flexível e a estrutura deve permitir modificações dinâmicas (pois a atividade de projeto é mutável), as operações de acesso de dados são longas, e o caráter iterativo e incremental do projeto exige que seja armazenado o histórico de modificações nos

dados. Uma solução, segundo o autor, era a adoção do banco de dados relacional, no qual eram descritos além dos elementos, os relacionamentos entre eles. Haynie também afirmou que os modelos semânticos de dados não eram uma nova tipologia, e sim uma especialização do modelo relacional de dados. Nos bancos de dados relacionais, um relacionamento é definido por uma tabela contendo um número fixo de atributos (colunas) e um número variável de matrizes unidimensionais (*tuples*). As várias matrizes unidimensionais são relacionadas entre si através de ponteiros chamados chaves primárias, compostas por uma ou mais das colunas de cada tabela (HAYNIE, 1983).

A abstração de dados proporcionada pelo modelo relacional foi um importante avanço, mas ainda era insuficiente para representar inequivocamente as informações de projetos de produtos. Para Eastman e Kutay, bancos de dados dedicados a representar projetos (ou bancos de dados de projetos), como quaisquer outros, são implementados com o propósito de integrar múltiplas aplicações em um ambiente comum, garantindo a consistência e a integridade dos dados durante as operações. Portanto, acessos concorrentes (quando vários usuários requisitam os mesmos dados) são ocorrências naturais, e devem ser controlados pelo sistema com naturalidade. Para isso, além da abstração de dados, outro tipo de abstração deveria ser incluída no modelo de dados – a abstração de operações sobre os dados.

O conceito de transação de dados é intimamente relacionado ao de abstração de operações: em vez de permitir operações arbitrárias sobre o banco de dados, são definidas coleções de operações – ou transações – que quando executadas mantém a integridade do banco de dados. A definição de integridade utilizada nos bancos de dados de aplicações de negócios também não era adequada ao ambiente de desenvolvimento de produtos. Se ela fosse aplicada, uma estrutura rígida de dados e atividades teria que ser imposta e, mesmo assim, um banco de dados de projeto passaria grande parte da sua vida em condição de inconsistência, já que a informação de um projeto só pode ser considerada completa quando ele está perto da sua conclusão. A natureza iterativa do projeto faz com que não apenas os dados, mas também a sua estrutura de armazenamento seja modificada durante as fases do desenvolvimento e, portanto, a condição de integridade deve ser relativa ao contexto.

No sistema proposto pelos autores, não haveria uma condição de integridade total, apenas integridades relativas, garantidas pelas próprias transações de dados. Durante a execução das transações de dados, a integridade poderia ser violada, já que uma transação pode significar a transição entre diferentes fases do projeto. Após a execução bem sucedida da transação, uma nova condição de integridade do banco de dados emerge e dependendo do resultado, uma série de outras transações pode ser requisitada para propagar o resultado da primeira sobre o restante do banco de dados (EASTMAN e KUTAY, 1989).

2.3.3 Orientação por objetos

Mais recentemente, uma nova tecnologia foi incorporada na criação de modelos de dados – a orientação por objetos. Essa tecnologia foi aplicada inicialmente em linguagens de programação de computadores, como o C++, no final da década de 1980. O objetivo era auxiliar os programadores a lidarem com a crescente complexidade dos programas de computador que até então eram baseados apenas em procedimentos. Como resultado, os programas passaram a ser organizados em conjuntos de objetos e interfaces para acesso aos dados. A orientação por objetos é um conceito de grande importância para a engenharia de produção e tem sido utilizado largamente em várias aplicações. Recentes esforços nessa área provaram que que a metodologia de modelagem orientada por objetos é capaz de digitalizar a informação sobre o produto em uma estrutura muito bem definida, onde os dados são mais facilmente acessados e modificados (YANG *et al.*, 2008). A tecnologia da orientação por objetos é composta por três princípios: encapsulamento, hereditariedade e polimorfismo (SCHILDT, 2002). O **encapsulamento** é o mecanismo que reúne os dados e os procedimentos para manipulá-los em um mesmo objeto, criando uma unidade de informação razoavelmente autônoma. Em um modelo de dados de produto, o encapsulamento poderia reunir vários tipos complexos de dados que representam um prisma, juntamente com as operações permitidas sobre estes dados (modificar uma das dimensões, por exemplo) em um único objeto. A **hereditariedade** é a propriedade que permite organizar hierarquicamente os objetos e reutilizar estruturas de dados previamente construídas através da sua especialização. No desenvolvimento de produtos, essa propriedade pode facilitar o processo de

derivação de componentes especializados a partir de formas básicas (sólidos geométricos transformados em componentes metálicos, por exemplo). Apesar de terem propriedades extras, os componentes derivados mantém as propriedades herdadas das formas básicas, para facilitar a edição do componente. Finalmente, o **polimorfismo** permite que uma única interface abstrata adapte-se a diversas situações, reduzindo o número de objetos que precisam ser criados. Há várias situações no desenvolvimento de produto onde essa característica é útil – um simples exemplo é permitir que múltiplos sistemas de medida (métrico e imperial, por exemplo) sejam utilizados sem que seja necessário reprogramar o objeto. O polimorfismo também pode ser mais complexo, reconstruindo totalmente a forma de um objeto quando o seu contexto é modificado. Atualmente, uma importante direção para a pesquisa sobre orientação por objetos no desenvolvimento de produtos é a apresentação dos dados do produto através de um formato padronizado. Um dos maiores avanços nesse sentido é a modelagem de dados baseada no padrão STEP.

2.3.4 STEP

STEP, acrônimo de *Standard for the Exchange of Product Model Data* (norma para transferência dos dados do produto) é o nome da norma ISO 10303, cujo objetivo é definir um formato neutro e interpretável para os dados do produto, durante todo o seu ciclo de vida, independente de sistemas específicos. A STEP é organizada em partes, os *Application Protocols* (protocolos de aplicação), ou APs, que definem padrões para estruturas de dados utilizadas por diferentes ramos da indústria. A norma também inclui uma linguagem formal para representação precisa e inequívoca dos dados do produto, chamada EXPRESS (SCRA-STEP, 2006).

As origens da STEP remontam a 1984, quando vários organismos de normatização nacionais reuniram-se para desenvolver uma única norma internacional de representação de modelos de produtos, cuja primeira versão foi publicada somente dez anos depois. A STEP foi um dos avanços mais significativos em direção à integração dos processos produtivos através dos modelos de produtos. A sua proposta de padronizar a transferência de todos os dados relativos ao produto estava muito à frente das demais normas de representação de dados, que ainda se concentravam

apenas na transmissão da informação geométrica (KRAUSE *et al.*, 1993). Gielingh observa que a importância de padronizações como a STEP, chamadas *Product Data Technologies* (PDT), reside no fato de não existirem aplicações de projeto auxiliado por computador que suportem todo o ciclo de vida de um produto. Ao contrário, os empreendimentos modernos são consórcios de companhias em colaboração, e o desenvolvimento de produtos é realizado em muitas aplicações diferentes, cada uma atuando em um escopo reduzido. Desse modo, apenas com a padronização de dados possibilitada pelas PDT pode ser viável a utilização de modelos integrados de produto (GIELINGH, 2008).

Os modelos de dados STEP são definidos utilizando a linguagem de descrição de meta-dados chamada EXPRESS, apresentada na norma ISO 10303-11:1994. A definição dos modelos pode ser realizada textualmente ou graficamente, com a extensão EXPRESS-G. A EXPRESS aplica o esquema semântico de representação de dados, baseado em entidades, atributos e relacionamentos, e também possibilita criar generalizações e restrições para os dados. Fowler atenta para o fato de a Express ser por vezes mal entendida, e lembra que (FOWLER, 1996):

- EXPRESS não é uma metodologia – ela pode ser utilizada em conjunto com várias metodologias de desenvolvimento;
- EXPRESS não é uma linguagem de modelagem completa – tipicamente um modelo de dados consiste na definição em Express complementada por definições em linguagem natural e em diagramas;
- EXPRESS não é uma linguagem de programação. Não é possível compilar o modelo de dados descrito, mas ele pode ser mapeado por diversas linguagens de programação;

Os constructos utilizados na definição de modelos de dados em EXPRESS são o *SCHEMA* – uma subdivisão funcional do modelo que permite a reutilização de informações entre modelos diferentes; o *TYPE*, que descreve tipos "primitivos" de dados, como inteiro, real, booleano, string, etc.; *ENTITY*, que representa as unidades básicas de informação, que compõem o *schema*; *SUBTYPE*, que estabelece relações hierárquicas entre entidades diferentes; *aggregations* (*SET*, *ARRAY*, *LIST*, *BAG*), que determinam coleções de tipos ou entidades; e também unidades algorítmicas:

FUNCTION, *PROCEDURE* e *RULE*, que são utilizadas para adicionar restrições adicionais ao modelo de dados (FOWLER, 1996).

Modelos de dados STEP são consideravelmente mais eficientes na transmissão de informações de projetos do que os modelos de dados utilizados nos CADs baseados em primitivos geométricos. Björk, por exemplo, propõe utilizá-los para permitir a transferência de informações a respeito dos espaços, das fronteiras espaciais e das estruturas de fechamento dos edifícios, que são utilizadas em várias aplicações de projeto e simulação (BJÖRK, 1992). Isso não seria possível sem um modelo de dados relacional, pois essas informações se baseiam justamente no relacionamento entre elementos do edifício. Um outro exemplo é a proposta de padronização de Fenves, que utiliza a EXPRESS para definir um núcleo comum de informações de projeto que viabilize a troca de dados entre vários setores da indústria. As entidades básicas do seu modelo semântico de dados são objetos genéricos capazes de descrever a forma e toda a informação complementar necessária para conduzir as atividades do projeto de produtos (FENVES, 2002).

Yang e outros autores, por outro lado, afirmam que muitas das pesquisas sobre a aplicação da STEP ainda são limitadas com relação ao principal objetivo da norma – o uso do padrão em um ambiente integrado. A maioria dos trabalhos apresenta a modelagem de produtos apenas em determinados estágios do desenvolvimento, enquanto no modo de produção atual, altamente cooperativo, diferentes fábricas necessitam trocar informações sobre vários tipos de produtos, descritos em diferentes modelos de dados. Em decorrência da variação existente entre os diferentes modelos, trabalho adicional é necessário para a conversão de dados, e em muitos casos essa conversão não garante a integridade da informação. Dados faltantes ou corrompidos tem grande impacto em empreendimentos de cooperação industrial, motivo pelo qual devem ser adotados modelos genéricos, capazes de descrever vários tipos de produtos. Os modelos de produtos desenvolvidos até então dão suporte a alguns processos da fabricação, mas não tem habilidade para suportar a totalidade do ciclo de vida dos produtos. Por isso, novas pesquisas são necessárias para desenvolver metodologias que permitam integrar efetivamente o desenvolvimento de produtos. A modelagem das características funcionais dos produtos deve se intensificar, e isso

exigirá uma abordagem cada vez mais abstrata e em alto nível para os modelos de dados. Também será necessário identificar meios para que o conteúdo semântico dos dados do produto seja armazenado e transmitido corretamente, para dar suporte a sistemas CAD mais inteligentes. Os processos de produção cada vez mais colaborativos e distribuídos também demandarão pesquisas no campo da utilização de tecnologias da *Web* para gerar e distribuir ambientes de desenvolvimento cooperativo de produto, e sobre a implementação da modelagem baseada na STEP para permitir a distribuição dos modelos entre os diferentes envolvidos (YANG *et al.*, 2008).

2.4 A modelagem de produto e o contexto da sua implantação

A implementação da modelagem de produto de maneira efetiva requer o estudo das aplicações computacionais que serão utilizadas e também do contexto no qual elas serão introduzidas. Esse contexto é formado pela organização operacional das empresas e pelo fluxo de informações que ocorre entre as diferentes etapas do desenvolvimento de um produto, e determina os requisitos a serem atendidos pelas ferramentas de informação utilizadas. A organização operacional define os envolvidos nas atividades e as suas responsabilidades na geração e no controle da informação. É essencial que a ferramenta de informação empregada registre esses dados para que as responsabilidades possam ser constantemente rastreadas. O fluxo de informações é um resultado direto da organização das operações de produção: para cada atividade do desenvolvimento de produtos deve ser identificado o conjunto completo de informações relacionadas semanticamente, que serão necessárias para as atividades subseqüentes (GRABOWSKI e ANDERL, 1983).

Krause e outros autores, em uma das mais influentes compilações sobre a modelagem de produto, definem uma estrutura semelhante, porém com diferentes nomenclaturas: os dois aspectos básicos a serem considerados pelas ferramentas de informação para a modelagem de produto são o modelo de produto em si, e as cadeias de processos produtivos envolvidas na sua fabricação. O modelo do produto é um repositório formado pela acumulação de toda a informação relevante sobre o produto, em uma estrutura de dados que forneça métodos de acesso adequados. Para auxiliar efetivamente nos processos de produção, o modelo do produto deve ser mais

do que uma representação estática do produto acabado. Ele deve conter tanto os resultados últimos como os intermediários, para que a seqüência de tomada de decisões que produziu a versão final do produto possa ser rastreada e analisada, permitindo que sejam aplicadas melhorias ao processo de produção. A informação armazenada no modelo de produto provém das cadeias de processos produtivos (*process chains*), que compreendem todas as atividades técnicas e gerenciais necessárias para transformar as idéias iniciais em produtos finais, durante todo o ciclo de vida do produto. Uma vez que toda a informação gerada em determinada etapa do ciclo de vida será utilizada em outra, a sua transmissão eficiente é essencial para o gerenciamento da cadeia, e quaisquer processos de tradução ou mudança de formato deveriam ser evitados, já que eles sempre trazem o risco de perda de parte do significado ou mesmo ocorrência de erros (KRAUSE *et al.*, 1993).

Mesmo um gerenciamento adequado de uma cadeia de processos perfeitamente ajustada pode não ser suficiente para garantir a competitividade de algumas empresas. Para as médias pequenas companhias, por exemplo, empregar recursos externos para alguns dos processos de produção é mais econômico e eficiente do que realizá-los internamente. O desenvolvimento distribuído de produtos é cada vez mais empregado nestes casos e, idealmente, a modelagem de produto deveria oferecer no mínimo dois graus de integração da informação nessa situação: entre as etapas de desenvolvimento realizadas dentro da organização, e entre os sistemas da organização e os sistemas utilizados por parceiros de desenvolvimento. Como qualquer ocorrência de conversão de dados na transmissão de informações nesses dois níveis de integração é indesejável, o modelo do produto deve adaptar-se a vários sistemas diferentes (YANG *et al.*, 2008).

Além de abranger uma vasta rede de interrelacionamentos entre as mais variadas tarefas técnicas, o desenvolvimento de produtos é também determinado pelos fatores humano, organizacional, as estratégias definidas pelas empresas para os produtos e as tecnologias que estão disponíveis. Deixar de considerar qualquer destes fatores durante a implementação da modelagem resulta em um investimento que gera baixo retorno ou até prejuízo. A implementação eficiente da modelagem de produto depende, por exemplo, da educação dos profissionais envolvidos: direcionar a sua

formação e investir em pesquisa básica é essencial, pois somente projetistas que compreendam as reais potencialidades e limitações da modelagem poderão desenvolver produtos de maneira eficiente (KRAUSE *et al.*, 1993). A questão do fator humano continua sendo importante tema de pesquisa relacionada à modelagem de produto, como se pode verificar no estudo realizado por Nilsson e Fagerström (2006), onde ressaltam a importância da consideração cuidadosa dos diversos envolvidos (*stakeholders*) e as suas diferentes necessidades por informação durante o desenvolvimento do produto.

Lars Hvam propôs uma abordagem para a implementação da modelagem de produto baseada na teoria dos sistemas, onde os processos que constituem as cadeias produtivas são considerados como as atividades de um sistema, que nesse caso é o desenvolvimento do produto. Na primeira etapa são analisadas as diferentes tarefas do sistema, o que servirá de base para determinar o grau ótimo de suporte das tecnologias de informação aplicadas. Essas análises enfocam três grupos de informação: a estratégia da companhia, o benefício econômico que pode ser potencialmente obtido pela implementação da modelagem de produto em cada uma das atividades, e finalmente as questões de representação de dados e estruturação do conhecimento sobre o produto. O resultado dessa fase fornece ao sistema de planejamento dos processos produtivos todas as informações necessárias para que se obtenha uma produção otimizada. Após concluídas essas análises, é iniciada a segunda etapa de implementação da modelagem, na qual são definidos os conteúdos e a estrutura dos sistemas de informação que darão suporte ao desenvolvimento do produto (HVAM, 2001).

2.5 Perspectivas para a modelagem de produto

A modelagem de produto não foi adotada em massa por todos os setores e atividades industriais, mas continua sendo tecnologia chave no desenvolvimento eficaz de produtos, e é essencial para as estratégias de competitividade das corporações. Documentos em papel continuam a ser substituídos por documentos eletrônicos, e estes continuam a ser substituídos por modelos de produtos. Ainda não foram desenvolvidas aplicações que dêem suporte a todo o ciclo de vida de um produto, e é

provável que esse nem seja um objetivo atualmente, visto que o desenvolvimento de produtos é um processo cada vez mais distribuído entre consórcios de organizações em cooperação. Neste sentido, considerar a possibilidade de transmissão de informações entre os processos e entre diferentes aplicações é determinante para o desenvolvimento eficiente de produtos (GIELINGH, 2008), e o principal veículo para a transmissão completa e inequívoca dessa informação ainda é o modelo de produto (YANG *et al.*, 2008).

3
Modelagem de Produto na Indústria da Construção

"Several organizations are developing computer programs capable of describing buildings at a detail allowing design and construction. These programs, consisting of a large database, routines for manipulation, analysis and document preparation, have the potential of replacing drawings as construction contract documents." — Charles Eastman, 1976.

A sequência de atividades realizadas durante o projeto e a construção de edifícios pode ser considerada um processo de desenvolvimento de produtos. Podem existir vários produtos intermediários durante o processo, na forma de entregas de projetos, mas o que mais interessa à modelagem de produto na construção é o produto que atende ao cliente final – a edificação concluída. Definir a construção como desenvolvimento de produtos permite introduzir na indústria várias das práticas já desenvolvidas para aumentar a eficiência na produção da indústria de manufatura. Os contextos das duas indústrias, porém, divergem largamente, e compreendê-los é essencial para que essa transferência seja bem sucedida. Projetos na indústria da construção são geralmente desenvolvidos por equipes fragmentadas, com pouca consideração das necessidades do cliente, e os produtos entregues normalmente estão acima do orçamento e do prazo definidos. Além disso, o projeto de um produto na indústria da manufatura dá origem a várias unidades indistintas, fabricadas em série, enquanto na construção o desenvolvimento de produto origina apenas uma unidade (TZORTZOPOULOS, 2004).

Outra especificidade da indústria da construção é a destinação principal dos investimentos. Pode-se classificar as tipologias produtivas em três grupos: as que concentram investimentos no desenvolvimento de novos produtos, as que concentram investimentos no desenvolvimento de melhores processos de produção, e as que concentram investimentos na manutenção de recursos (pessoas ou equipamentos) – que é o caso da indústria da construção. Por investir pouco em desenvolvimento de produtos e processos de fabricação, a indústria da construção não se beneficia dos avanços nas técnicas de gestão da produção tanto quanto outros setores industriais, em especial o de manufatura. Além disso, enquanto outros setores concentram

grandes quantidades de investimento antes mesmo de qualquer comercialização, a indústria da construção é essencialmente dependente da contratação formal do serviço, o que em geral limita a capacidade de escolha e determinação de estratégias próprias das empresas (WORTMANN, 1992).

A construção demanda ferramentas de informação adequadas ao desenvolvimento *one-of-a-kind*, no qual situações diferenciadas e imprevisíveis invariavelmente emergem a cada projeto. Os requisitos para as ferramentas são decorrência da necessidade de fabricação de produtos únicos, através de processos únicos, por um grupo de parceiros configurado unicamente para um projeto específico. Esta situação não é favorável ao desenvolvimento de sistemas de informação, porque em geral os desenvolvedores procuram por procedimentos que se repitam, e estruturam as aplicações a partir de algoritmos que respondem a eles. Associada a esta condição desfavorável, existe a resistência à adoção de tecnologias por parte dos atores da indústria da construção. Segundo Turk, em projetos típicos na construção, colaboram parceiros com diferentes níveis de proficiência em tecnologias de informação, e normalmente o denominador comum é nível menor. A média de domínio das tecnologias de informação por equipes no setor da construção também é considerada mais baixa do que em outros setores da indústria. A construção é um ambiente ruim para a transferência de tecnologia, por ser comparativamente mais conservadora, com pouca inovação no corpo central de conhecimentos técnicos, e não incentivar a educação continuada como prática comum. Finalmente, os profissionais da construção – particularmente os projetistas e consultores – trabalham em vários projetos simultaneamente, e a introdução de uma nova tecnologia por um determinado projeto gera a necessidade de trabalharem com dois tipos de ferramentas ao mesmo tempo (TURK, 2006a).

Nascimento e Santos classificam em quatro grupos as barreiras para o uso da tecnologia da informação na construção civil. Aspectos profissionais estariam no primeiro grupo: profissionais contratados com menos exigências, metodologias de trabalho diferentes, gerentes pouco capacitados, modo de trabalho individual e resistência a mudanças. No segundo grupo, barreiras ligadas aos processos gerenciais: operações que não se utilizam das tecnologias mais recentes, falta de padronização na

comunicação e métodos de gestão ultrapassados. Um terceiro grupo é formado por barreiras ligadas às empresas e à cultura corporativa: baixo investimento em TI, falta de confiança nos resultados obtidos pela TI e falta de treinamento em tecnologias. Finalmente, um quarto grupo de barreiras, diretamente ligadas aos próprios aspectos tecnológicos: segurança dos dados contra intrusões e violações, conexões de baixa velocidade, custos de aquisição e manutenção dos equipamentos (NASCIMENTO e SANTOS, 2002).

Como consequência das suas características diferenciadas e da resistência à adoção de tecnologias de informação, a indústria da construção é considerada tecnologicamente atrasada em relação aos demais setores, com baixos índices de produtividade mesmo em países industrializados (LAM *et al.*, 2005; BÉDARD, 2006). Pesquisas realizadas por Wang e outros autores revelaram que o atraso dos processos de execução de obras em decorrência de falhas na documentação do projeto é considerado fato comum para a indústria da construção britânica. Um terço das obras daquele país sofrem atrasos ou excedem os orçamentos iniciais em virtude de informações incorretas contidas nos desenhos e documentos (WANG et al., 2006).

No Brasil, os métodos empregados na construção de edificações mostram-se tecnologicamente defasados mesmo em comparação com outros setores da própria indústria da construção, como a construção de obras de infra-estrutura (NOVAES, 1996). A defasagem tecnológica do processo construtivo resulta em maiores custos, desperdício de materiais, baixa produtividade e produtos de má qualidade (MELHADO e AQUINO, 2001). Souza, estudando os processos empregados por empresas de construção, constata que determinadas fases da execução da obra, como as alvenarias de vedação, podem apresentar um índice de desperdício de materiais de até 40%. O autor defende que projetos com mais qualidade podem racionalizar o consumo de materiais, reduzindo a extração de matérias primas (SOUZA, 2005).

Wortmann observou que a maioria dos aspectos cobertos pelas teorias de gestão da produção aplicavam-se somente à fabricação em série de produtos anônimos. Segundo ele, os sistemas de informação utilizados nesse tipo de fabricação geralmente assumem que a informação completa sobre o produto é um pré-requisito

para o início da fase de produção. A gestão da fase de produção, por sua vez, é considerada apenas uma questão de tomar decisões racionais baseadas em relatórios de acompanhamento e totalização. Na produção de artefatos únicos, entretanto, a informação completa, quando existe, fica disponível apenas na conclusão do empreendimento, e a gestão da produção deve motivar os profissionais envolvidos para que atuem cooperativamente, compensando esta desvantagem. Os sistemas de informação para o desenvolvimento de produtos únicos devem considerar esse aspecto, e proporcionar meios para a rápida atualização da informação entre todos os envolvidos quando ocorrem solicitações por parte dos clientes (WORTMANN, 1992). Eastman e Kutay, em consideração sobre estruturas de dados para sistemas CAD na indústria da construção, fazem a mesma observação sobre o caráter inerentemente incompleto da informação do projeto de edifícios (EASTMAN e KUTAY, 1989).

Outra característica da produção na indústria da construção é a complexidade do produto. Edificações são compostas por um milhares de partes distintas, que podem ser organizadas em várias composições, dependendo da função e dos processos construtivos. Por isso, é comum haver pequenas diferenças mesmo entre instâncias de uma parte que é utilizada repetidamente. Todas essas partes devem ser modeladas para que a integridade da informação seja mantida, assim como ocorre em produtos de manufatura. Porém um projeto de edificação tipicamente concentra-se na integração entre diferentes sistemas e composições, enquanto no desenho industrial ele concentra-se na otimização de componentes individuais para a produção em série. Embora os conceitos centrais sejam os mesmos, as funcionalidades e interfaces de um sistema para modelagem de edifícios são muito diferentes dos utilizados na manufatura. Por exemplo, mesmo que a modelagem de sólidos possa permitir a definição intuitiva e precisa da forma de uma peça mecânica, não é óbvio como essa funcionalidade deveria ser empregada no projeto de edifícios, que envolvem composições de grande números de partes. Revisar e modificar um modelo composto por sólidos geométricos representando partes detalhadas individualmente pode ser uma atividade mais demorada e sujeita a erros do que fazê-lo usando desenhos em papel (SACKS *et al.*, 2004). Isso pode ser ilustrado pelo recente trabalho apresentado por Marcos e outros autores, no qual são avaliadas as condições de acessibilidade em

uma habitação utilizando o *software* CATIA (MARCOS *et al.*, 2007). As funcionalidades de identificação de conflitos espaciais e visualização tridimensional oferecidas pelo programa foram úteis na verificação da adequação do produto e produziram uma análise adequada, mas a criação e a modificação do modelo utilizado para a simulação foram dificultadas porque o CATIA não oferece funcionalidades necessárias para operações comuns no projeto de edifícios, como mover janelas e portas ou visualizar o edifício em planta.

Como resultado dessas especificidades, surgiu um ramo da informática especializado na criação de aplicações para a indústria da construção. Ziga Turk propôs a definição formal do corpo de conhecimentos que durante os anos foi conhecido como *construction informatics*, *computing in civil engineering, construction information technology* ou ainda *information and communication technology in construction* e outros nomes. A informática na construção é um ramo científico com uma ênfase própria sobre as teorias de informação e computação, orientado para a solução dos problemas específicos oriundos dos processos de projeto e construção de edifícios, situado entre a pesquisa científica e a solução de problemas de engenharia (TURK, 2006b).

3.1 Origens da modelagem de produto na indústria da construção

A idéia da modelagem de produto na indústria da construção é quase tão antiga quanto os primeiros sistemas CAD desenvolvidos pelo grupo de Douglas Ross no MIT, no início da década de 1960. Vários dos conceitos fundamentais da aplicação da modelagem no desenvolvimento de produtos na construção foram apresentados ainda durante a década de 1970. Gingerich, em 1973, observou que os programas de necessidades, sistemas e materiais utilizados nos edifícios estavam evoluindo muito rapidamente, e que a coordenação do projeto tornava-se cada vez mais complexa. Uma resposta a essa situação seria o uso de abordagens mais eficientes para o computador, que integrassem as tarefas que até então eram executadas por aplicações isoladas. Ele apresentou um protótipo de um sistema para a fase de definição da volumetria no projeto arquitetônico, baseado em duas interfaces: uma bidimensional, onde os elementos do projeto eram inseridos, e outra tridimensional,

onde poderiam ser visualizados e modificados. As interfaces eram integradas e atualizavam-se automaticamente após as modificações no projeto. Uma terceira interface foi prevista pelo autor, que permitiria detalhar o projeto inserindo materiais, acessórios, portas, janelas e sistemas estruturais (GINGERICH, 1973).

A então nascente tecnologia de bancos de dados foi por muito tempo indispensável para o desenvolvimento de sistemas de modelagem de edifícios, pois oferecia a possibilidade de estruturar dados mais adequadamente, e recuperá-los mais rapidamente do que com a utilização de arquivos sequenciais. Em 1975, Charles Eastman cunhou a expressão *Building Description System* (BDS) para designar os CADs que se baseavam não em desenhos, mas sim em estruturas de dados contendo informação geométrica associada a atributos diversos, capazes de representar mais adequadamente os elementos de um projeto. BDSs foram definidos pelo autor como grandes sistemas de informação, com rotinas para entrada, manutenção de dados, processamento de análises diversas e geração automática de relatórios. Desenhos técnicos, como os necessários para a construção do edifício, eram apenas mais um tipo de relatório, descrito graficamente. Eastman propôs que essa forma de representar edificações poderia tornar-se a principal documentação utilizada na indústria da construção (EASTMAN, 1976).

Os modelos utilizados nestes sistemas precisariam ser completos e coerentes, representando tanto os elementos do edifício como os seus arranjos. Dada uma completa representação tridimensional do artefato sendo projetado, o projetista poderia ter certeza de que todas as projeções bidimensionais geradas a partir dela seriam consistentes. As informações sobre a forma dos objetos poderiam ser integradas com informações funcionais e de desempenho, então aplicações poderiam acessar e manipular os dois tipos de dado, sem traduções manuais que são costumeiras quando se utiliza os desenhos. Aplicações utilizando modelos de edifícios poderiam fazer verificações de conformidade, avaliar o projeto estrutural, térmico ou de outras propriedades, estimar custos ou adicionar detalhes padronizados. Outras aplicações poderiam gerar visualizações, projetos para construção e controle numérico para produção de peças. Muitas outras aplicações poderiam ser imaginadas para os mais diferentes segmentos do projeto (EASTMAN e HENRION, 1977).

Apesar das vantagens teóricas e de apresentações ocasionais de sistemas protótipicos, a modelagem de produto foi ainda menos adotada na indústria da construção do que fora registrado em outros setores da indústria. Kalay, em 1985, observa que o rápido desenvolvimento tecnológico havia gerado uma crença na possibilidade de aumento de produtividade e economia de recursos pelo uso do computador. Como a produção de edifícios cada vez mais complexos envolvia uma quantidade crescente de recursos físicos e informações, imaginou-se que os processos da indústria da construção deveriam seguir a mesma tendência de integração e automação observada na indústria da manufatura. Entretanto, mesmo sendo desenvolvidos havia duas décadas, os sistemas CAD permaneciam gerando um impacto apenas marginal no processo de projeto de edificações. Na engenharia elétrica, um dos principais campos de desenvolvimento do CAD nos primeiros anos da tecnologia, a adoção e a influência fora imediata: permitiu que os projetistas aumentassem a complexidade dos circuitos integrados em várias vezes, reduzindo significativamente o tempo de desenvolvimento. Para Kalay, a falha do CAD em melhorar as práticas de projeto de edificações e os seus produtos era resultado principalmente do papel dado aos computadores no processo de projeto: mais de 90% dos sistemas instalados ao redor do mundo, até então, eram utilizados apenas para desenho (*drafting*). Para ele, digitalizar desenhos não era necessariamente uma etapa essencial no progresso de um produto da concepção à produção, mas apenas uma forma de comunicação do resultado esperado para uma etapa. Para que os computadores fossem empregados de maneira mais efetiva no processo de projeto de edificações, a sua capacidade deveria ser desenvolvida da mera descrição de informações geométricas para a simulação de decisões de projeto. Era necessário incluir o significado técnico dos elementos do projeto e apoiar o processo de análise das decisões projetuais através de um conjunto de regras e procedimentos que fossem capazes de extrair informação relevante do modelo. Deveria ser possível inferir informações que não fossem explicitamente modeladas, e também selecionar ações que modificassem o modelo da maneira desejada (KALAY, 1985b).

Em uma concisa revisão sobre o desenvolvimento do CAD na indústria da construção durante a década de 1980, Eastman relata que os objetivos originais que

justificaram a tecnologia continuavam longe de ser implementados na prática. A década de 1980 observou a popularização do *Personal Computer*, o PC, em substituição aos minicomputadores utilizados na década anterior para o desenvolvimento de vários sistemas experimentais e comerciais. Também foi nesta década que o modelo de dados proprietário da Autodesk, o DXF, tornou-se a formato mais utilizado para a troca de informações na indústria (KALAY, 1985b). Embora em teoria o nível tecnológico da computação já possibilitasse a implementação de várias das propostas iniciais do CAD para projeto de edificações, o desenvolvimento de edifícios no computador continuava centrado em desenhos. Além disso, Eastman observa que a maioria dos escritórios que haviam adotado o CAD o fizeram primeiramente por pressão dos clientes, que procuravam por empresas que transmitissem uma imagem de modernidade. Mesmo o benefício da agilização de tarefas repetitivas proporcionado pela digitalização dos desenhos era considerado apenas em segundo plano. De modo geral, a grande falha na implementação do computador no processo de desenvolvimento de produtos na construção continuava sendo a restrita utilização do conceito de modelagem de produto (EASTMAN, 1989).

CADs comerciais para modelagem de edifícios foram disponibilizados já no início da década de 1980. Essa foi a mesma época do surgimento dos CADs comerciais de desenho. Alguns textos recentes sugerem que as versões comerciais dos CADs de modelagem, atualmente conhecidos como BIM CADs evoluíram a partir dos CADs comerciais de desenho. Tse e outros autores, porém, afirmam o contrário: a primeira versão do Allplan, da alemã Nemetschek, data de 1980 (NEMETSCHEK, 2008). A empresa húngara Graphisoft lançou em 1984 o Radar CH, que na sua segunda versão (em 1986), passaria a chamar-se ArchiCAD (GRAPHISOFT, 2008a). Ambos estavam bastante a frente do seu tempo, considerando que a primeira versão do Autocad é de 1983 e a do MicroStation data de 1984. Não houve, portanto, relação de evolução entre os softwares. A representação de edifícios por modelagem e a representação por desenhos foram duas abordagens diferentes adotadas pelas empresas desenvolvedoras desde o início da disseminação dos CADs comerciais. Porém, a proposta da modelagem de edifícios estava muito distante da capacidade de *hardware* disponível nos computadores da época, e os CADs baseados em primitivos geométricos

e a representação do edifício por meio de desenhos acabaram por tornar-se o padrão para o uso do computador na indústria da construção (TSE *et al.*, 2005).

No início da década de 1990, CADs de modelagem de edifícios já eram comercialmente disponíveis há dez anos. O ArchiCAD, por exemplo, estava na quarta versão e a sua interface de modelagem de edifícios e geração automática de documentação já era perfeitamente reconhecível para os usuários atuais (VÉRTESI *et al.*, 1991). A modelagem de edifícios ficou disponível para o público geral inicialmente nos sistemas Macintosh, e para o projeto de pequenas construções. Essa situação começaria a mudar com a exigência por maior qualidade e processos mais eficientes na indústria da construção, que fez ressurgir o interesse pela modelagem de produto. Eastman atentou então para o fato das idéias estarem sendo redescobertas ou até reinventadas, e que muito do insucesso da aplicação da tecnologia desde a década de 1970 era resultado da falta de compreensão dos seus objetivos e potencialidades para a indústria da construção (EASTMAN, 1992).

3.1.1 BIM

Atualmente, o novo nome da antiga proposta de modelagem de produto na construção é BIM, acrônimo de *Building Information Modeling* (IBRAHIM *et al.*, 2003). O termo BIM foi criado pela empresa americana Autodesk em meados dos anos 1990 para promover o seu novo CAD, o Revit. A idéia era reunir em um único conceito (de *marketing*, inclusive) o conjunto de funcionalidades integradas oferecidas pelo novo produto. Em essência, o Revit é um CAD de modelagem de edifícios, assim como ArchiCAD e Allplan já eram mais de dez anos antes. Porém o termo BIM mostrou ter um forte apelo comercial, e logo foi adotado pelas demais fabricantes como estratégia de mercado para divulgar os seus próprios CADs de modelagem de edifícios. Definir BIM como um tipo de *software*, porém, reduz muito o seu significado, que é derivado da longa tradição de utilização do computador como suporte ao projeto, apresentada nos itens anteriores. Neste trabalho, o termo BIM é utilizado no sentido proposto por Eastman e outros autores (EASTMAN *et al.*, 2008), que na verdade é uma compilação dos princípios da modelagem de produto na construção, desenvolvidos a partir da década de 1970.

Seja a modelagem de produto na construção chamada BIM, prototipação de edifício, modelagem de edifício, construção virtual ou qualquer outro nome, é atualmente considerada um catalisador para a adoção das práticas integradas de projeto e da construção enxuta. A sua utilização tem demonstrado significativas vantagens sobre processos tradicionais, mesmo em situações de integração limitada (por exemplo, apenas entre o projeto arquitetônico e o estrutural). No futuro, espera-se que uma integração mais ampla dê origem a novas oportunidades de negócios e melhore a produtividade da indústria da construção, que é sabidamente inferior aos outros setores da indústria. Contratantes e profissionais do setor apenas começaram a compreender as novas possibilidades oferecidas (NIBS, 2007).

3.2 Escopo da BIM

Na indústria da manufatura, a modelagem de produtos surgiu para integrar a informação em todos os processos do ciclo de vida de um produto. O seu campo de estudo, portanto, abrange tudo que está relacionado com qualquer atividade entre a concepção e a disposição final do produto. Do mesmo modo, a BIM abarca um amplo espectro de conceitos, atividades, técnicas, ferramentas e atores, reunidos em relacionamentos complexos e distribuídos por todas as atividades inerentes à indústria da construção. Estudos sobre a BIM podem incluir trabalhos com abordagens tão diversas quanto a definição fenomenológica do termo “modelo” (TURK, 2001) e a estruturação lógica do seu armazenamento em disco (HANNUS, 1991).

Não obstante a sua amplitude, a BIM pode ser mais facilmente compreendida se for abordada em diferentes níveis de abstração, sendo níveis mais altos relacionados com o contexto da aplicação da tecnologia, e os mais baixos relacionados com os aspectos mais técnicos das suas ferramentas. Por exemplo, A iniciativa de regulamentação da modelagem de produtos para a indústria de obras de infra-estrutura nos Estados Unidos, a *National Building Modeling Information Standard* (NBIMS), adota um esquema de abstração em três níveis: A BIM é entendida como um produto, como uma ferramenta e como um processo. Como um produto, a BIM refere-se ao modelo da edificação, ou seja, uma entrega do processo de projeto baseada em padrões abertos e criada por ferramentas de informação. Como ferramenta, a BIM

refere-se às aplicações que interpretam o modelo da edificação e agregam informações e representações a ele, chamadas *BIM authoring tools*. Por fim, a BIM é entendida como um processo colaborativo formado por atividades desenvolvidas durante todo o ciclo de vida da edificação (NIBS, 2007).

Interpretar a BIM como processo, ou seja, a partir de um nível de abstração mais alto, é essencial para a sua efetiva compreensão, já que qualquer modelagem de produto tem como prérequisito integrar diferentes fases do desenvolvimento de um produto. Para este trabalho porém, os níveis mais baixos de abstração tem uma importância central, pois são cruciais para o acesso aos modelos de edifícios. Além disso, ajustar os níveis de abstração para que enfoquem os aspectos mais técnicos, mesmo quando são relacionados ao contexto da aplicação da modelagem, é mais útil para os objetivos desta dissertação.

Por este motivo é proposta uma classificação em quatro níveis de abstração: metamodelagem, modelagem, modelo, e objetos – as partes que compõem o modelo. No nível mais alto, o da metamodelagem, figuram as questões sobre modelos conceituais, interoperabilidade de aplicações e os impactos da tecnologia sobre a indústria, por exemplo. No nível da modelagem são abordadas questões relacionadas à criação dos modelos, e consequentemente as funcionalidades e interfaces das aplicações CAD que realizam a modelagem de produto, também chamados BIM CAD ou *BIM-based CAD* (IBRAHIM *et al.*, 2004). No nível do modelo são enfocadas as relações semânticas entre os diferentes objetos que o compõem. Finalmente, no nível mais baixo, o dos objetos, são abordadas as questões sobre a funcionalidade das partes que compõem o modelo, como inteligência contextual, comportamento, atributos necessários para a descrição de elementos construtivos, entre outros.

Cada nível de abstração proposto inclui diversas características importantes para a tecnologia BIM, mas é possível perceber na literatura sobre o tema que existem conceitos centrais que podem ser atribuídos para cada nível, quando a BIM é enfocada a partir de um ponto de vista mais técnico. Na metamodelagem, o conceito principal é a interoperabilidade entre aplicações; na modelagem, a consistência da informação; no modelo, a estruturação semântica; e nos objetos, o comportamento. Os trabalhos

consultados, embora utilizem terminologias diferentes para os conceitos, não atribuem o termo modelagem (ou equivalente) a abordagens de desenvolvimento de produtos que não apresentem estas quatro qualidades.

Nas seções seguintes será apresentada uma rápida revisão das principais características da tecnologia de modelagem do edifício, baseada na estruturação dos quatro níveis de abstração proposta. Esta revisão foi baseada em definições e observações sobre o tema apresentadas em trabalhos científicos a partir da década de 1970. Muitos deles, portanto, ainda não se referiam à modelagem de produto pelo termo BIM, mas isso não afeta a sua relação com o desenvolvimento atual do tema. Não é intenção deste trabalho produzir uma revisão exaustiva nem absoluta: os artigos citados destacam-se por terem exercido influência considerável sobre o campo de pesquisa – verificada pelo número de trabalhos que os citam como referência – ou por conterem uma abordagem didática diferenciada. Para uma revisão mais aprofundada, sugere-se que o leitor consulte os dois trabalhos mais extensos sobre modelagem de produto na construção, os quais cobrem os quatro níveis de abstração mencionados: *Building Product Models: Computer Environments Supporting Design and Construction* (EASTMAN, 1999) e o mais recente *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors* (EASTMAN *et al.*, 2008).

3.3 Objetos

O conceito de objeto é vital para a modelagem de edifícios, e é possível perceber o seu delineamento a partir da década de 1970, em vários trabalhos citados nessa dissertação. Objetos são unidades de informação criadas para representar os diferentes elementos que constituem um projeto de edificação, incluindo as suas características e relacionamentos com outros objetos (HANNUS, 1991). Eles contém informação suficiente para permitir vários tipos de análises e representações (EASTMAN, 1992; IBRAHIM *et al.*, 2004). Os principais tipos de objetos são os que representam elementos construtivos, como paredes, colunas, vigas, janelas, etc., mas também há objetos que representam espaços, zonas, mecanismos e até as simbologias

utilizadas nos desenhos, como cotas, indicações, níveis, entre outros (EASTMAN, 1976).

Diferentes coleções de objetos podem ser desenvolvidas para agilizar o processo de projeto ou representar mais fielmente os elementos utilizados. Escritórios podem desenvolver padrões ou módulos contendo informações utilizadas regularmente e fabricantes podem oferecer objetos representando seus produtos, do mesmo modo que é feito com blocos no Autocad, porém contendo mais informações do que simples representações bidimensionais (EASTMAN, 1976). Ibrahim e Pentilla citam a possibilidade de disponibilização de catálogos de objetos na Web, ou bibliotecas públicas de objetos, permitindo que os projetistas insiram rapidamente informações geradas pelos fabricantes de materiais (IBRAHIM *et al.*, 2004; PENTTILA, 2005). Essa operação poderia agregar automaticamente ao projeto não só a forma e a representação dos elementos construtivos, como também o custo, prazo de entrega, desempenho, instruções para montagem ou construção, e também para operação, manutenção e disposição dos materiais. Essa informação poderia ser extraída posteriormente, por aplicações de análise ou planejamento da construção, por exemplo. Embora algumas bibliotecas de objetos já sejam disponíveis – principalmente para peças de mobília – a informação contida ainda é limitada, e comumente disponibilizada em formatos proprietários, que não podem ser utilizados em todos os BIM CADs.

Os objetos são estruturas de dados conhecidas como *classes* (como descrito na subseção 2.3.3). Classes, em desenvolvimento de softwares, são unidades de informação constituídas por dados e procedimentos para processar esses dados. Na modelagem de produtos, convencionou-se chamar os dados de parâmetros do objeto, de onde vem os nomes objetos paramétricos e modelagem paramétrica. Os procedimentos são conhecidos como comportamento do objeto.

3.3.1 Parâmetros

A informação a respeito do elemento construtivo a ser representado é armazenada em diferentes parâmetros que podem ser combinados pelo usuário para produzir diferentes respostas. Ibrahim e outros autores afirmam que há dois tipos

básicos de parâmetros: os que armazenam informação sobre a forma dos elementos – como posição, dimensões ou transformações geométricas – e parâmetros que armazenam características funcionais dos elementos, como material, especificações, requisitos legais, procedimento de montagem, preço, fabricante, distribuidor, etc. (IBRAHIM *et al.*, 2003).

Parâmetros geométricos e parâmetros funcionais podem armazenar todas as informações disponíveis sobre um elemento construtivo, mas eles não determinam necessariamente a forma com que são representados. Eastman afirma que durante algum tempo acreditou-se que operações de corte realizadas nos sólidos geométricos construídos a partir dos parâmetros dos objetos seriam suficientes para gerar a documentação para a construção. Plantas, cortes e elevações eram consideradas apenas uma questão de situar apropriadamente um plano e então extrair os pontos das superfícies dos sólidos que estivessem contidos nele. Porém, apenas o resultado dessa operação não gera informação suficiente para as representações utilizadas na construção de edifícios, que são em grande parte simbólicas. Desse modo, as criação das representações de edifícios partem da geometria resultante da operação de corte, mas devem ser complementadas por características determinadas pelas convenções de desenho técnico. Essas convenções incluem, por exemplo, espessuras e tipos diferentes de linhas, hachuras, indicações textuais ou pictóricas. Produzir esse tipo de informação a partir somente de operações de corte de sólidos geométricos é praticamente impossível, ou exigiria um trabalho adicional maior do que o necessário para produzir desenhos em vez de utilizar modelos. Um exemplo disso são as portas, cuja representação simbólica varia de acordo com o tipo de documento desejado, ou indicações de pontos elétricos, que são representados em planta como um símbolo, e não como tomadas em corte. Para se adequar a esta situação, os CADs para modelagem de edifício precisam de um terceiro conjunto de parâmetros, relacionados às diferentes representações possíveis para um determinado elemento construtivo. Estes parâmetros indicam cores, linhas, hachuras, caracteres, simbologias padronizadas para a complementação das representações, a partir da geometria resultante das operações de corte (EASTMAN, 1991). Nos BIM CADs esses parâmetros

são acessados a partir da janela de configuração dos objetos, como mostrado na figura 3.01.

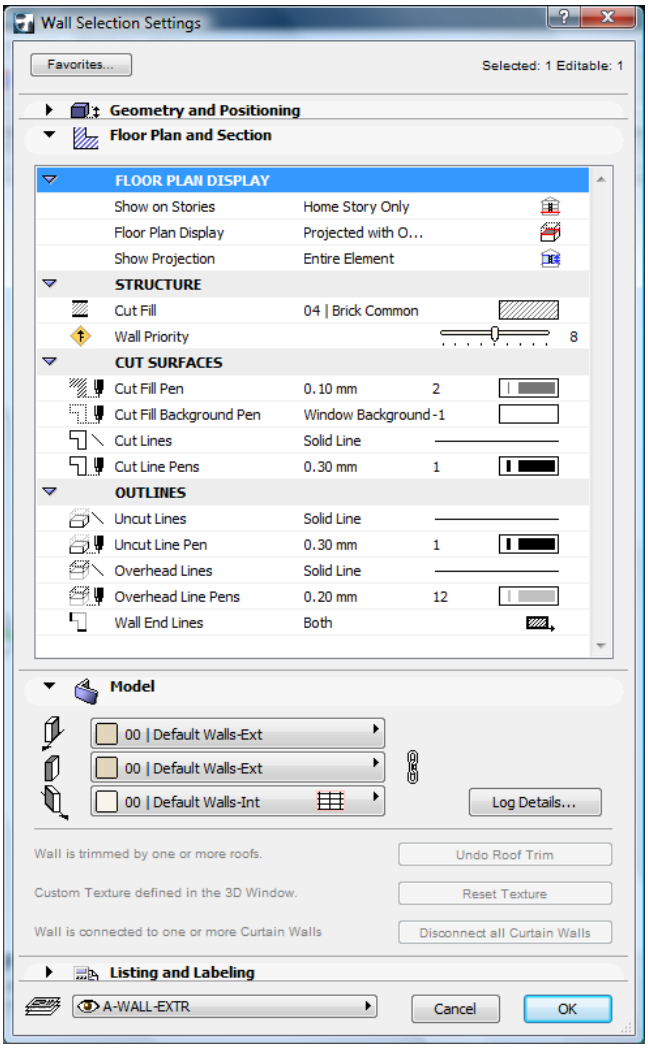


Fig. 3.01: janela de configuração dos parâmetros de representação de um objeto "parede" no ArchiCAD.

Em algumas situações, como detalhamentos em grande escala, pode ser necessário substituir completamente a geometria resultante da operação de corte por um conjunto arbitrário de primitivos geométricos, que representam melhor a informação ou tornam o trabalho de desenho mais fácil (EASTMAN, 1992). Essa substituição depende da intenção da documentação resultante, e da fase em que se encontra o projeto. Por exemplo, não é prático modelar inteiramente os sistemas hidráulico-elétricos de um edifício apenas para que as tubulações apareçam cortadas nos detalhes dos *shafts* no projeto arquitetônico. Essa informação, embora correta,

não agrega valor suficiente para compensar o complicado trabalho de modelagem. Por outro lado, o trabalho de modelagem dos elementos dos dutos de ar pode compensar em projetos complementares, onde pode facilitar a avaliação das diversas soluções possíveis, permitir a identificação de conflitos espaciais, e produzir automaticamente a documentação para a montagem dos sistemas.

3.3.2 Comportamento

Não há limitações para o número de parâmetros que podem ser incluídos em um objeto, mas algumas situações de projeto não podem ser resolvidas apenas com dados, por mais bem estruturados que sejam. São necessários também procedimentos (EASTMAN, 1992). Procedimentos são algoritmos descritos em linguagens de programação estruturadas que são encapsulados nos objetos paramétricos e originam os seus diferentes comportamentos. Os comportamentos são então ativados diretamente pelo usuário, ou indiretamente por rotinas do BIM CAD, gerando como resultados informações que podem ser utilizadas em análises ou para criar representações (EASTMAN, 1991). Lee e outros autores afirmam que o comportamento do objeto é a capacidade de responder a estímulos externos e internos. Os estímulos externos são gerados por uma situação comum a todos os objetos, como uma mudança de variáveis globais. Esta variável é então passada para todos os objetos, podendo sobrepor seus parâmetros originais. Os estímulos internos são causados pela modificação de atributos do próprio objeto (LEE *et al.*, 2006).

Por exemplo, o objeto *door* do ArchiCAD responde a modificação do parâmetro altura da porta – um estímulo interno – atualizando automaticamente a representação em planta do objeto. O objeto também pode responder a um estímulo externo, como a redefinição da escala de representação da planta (uma variável global). A variável global “escala” é passada a todos os objetos presentes (os que são visíveis), o que provoca uma reorganização e seleção automática de parâmetros de cada um e a consequente atualização das representações resultantes. O resultado de uma modificação de variável global em objetos é mostrado na figura 3.02: o comportamento dos objetos gera uma representação mais simplificada e ajusta hachuras e o símbolo.

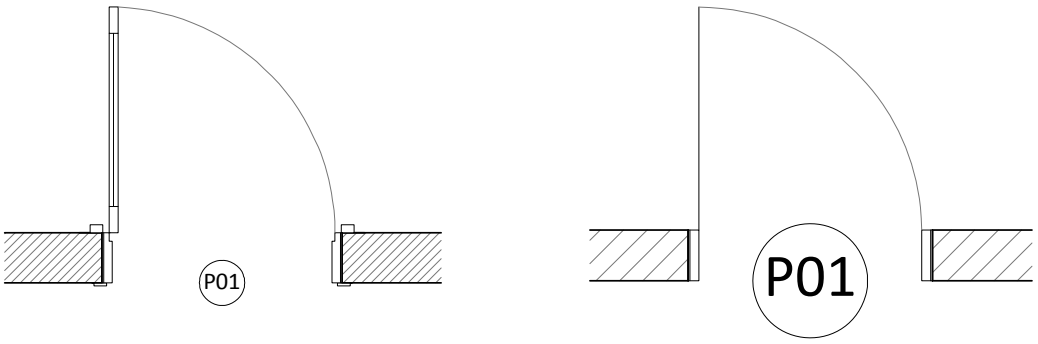


Fig. 3.02: atualização automática da representação do objeto quando a escala de representação em planta é modificada de 1:20 para 1:100, no ArchiCAD.

A natureza da vista selecionada pelo usuário (planta, corte, elevação, perspectiva) também pode ser considerada uma variável global, e gerar diferentes vistas é um estímulo externo ao objeto. Ao mudar da representação em planta para um corte, por exemplo, o BIM CAD passa a todos os objetos presentes a variável global que determina o tipo de vista no qual devem ser representados. O comportamento de cada objeto então responde processando os parâmetros adequados e produzindo uma nova instância de representação automaticamente (EASTMAN *et al.*, 2004). Na figura 3.03 são mostradas quatro instâncias de representação originadas por um objeto *door* do ArchiCAD: planta, elevação, corte, e dois tipos de perspectiva – uma detalhada, para ser utilizada como uma maquete eletrônica realista, e uma simplificada, que pode ser exportada para um *software* de análise de desempenho.

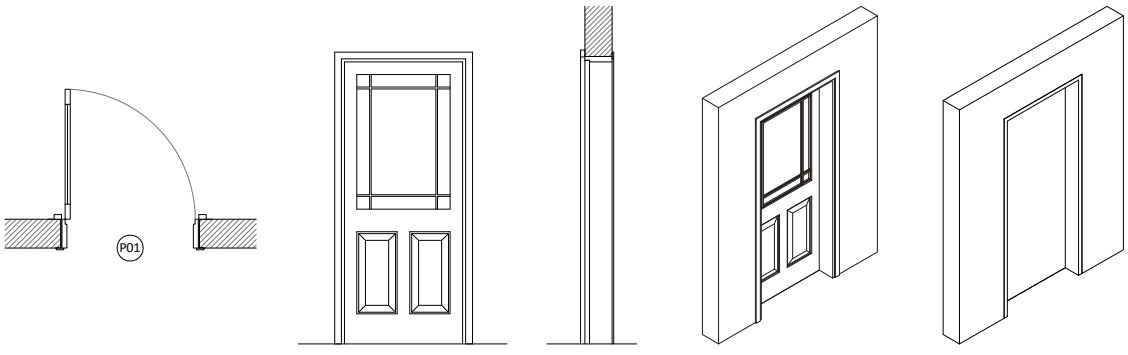


Fig. 3.03: diferentes representações gráficas instanciadas a partir de um mesmo objeto door do ArchiCAD

Embora as diferentes instâncias de representação sejam configuradas pelos diferentes parâmetros definidos pelo usuário, é o comportamento dos objetos que as insere no local e no momento adequado, automaticamente. Essa característica,

também conhecida por "sensibilidade ao contexto" (ou *context aware*), é o que permite ao BIM CAD extrair diferentes informações dos objetos, combinando os seus parâmetros de acordo com situações específicas.

Uma importante funcionalidade derivada da sensibilidade ao contexto é o ajuste automático do nível de detalhe da representação de um objeto de acordo com a fase do desenvolvimento do edifício. Durante o projeto, a informação torna-se articulada incrementalmente, e não seria natural nem conveniente exigir que o usuário fornecesse todos os atributos do objeto durante a sua primeira inserção (EASTMAN, 1976). Além disso, o projeto de edifícios é um processo multidisciplinar, que envolve participantes, conhecimentos e informações de vários domínios. Cada usuário envolvido na modelagem possui uma visão própria sobre a solução dos problemas projetuais, decorrente da sua profissão. Portanto, a modelagem requer uma multiplicidade de visões, cada uma enfatizando informações específicas de disciplinas ou fases do projeto, em diferentes níveis de resolução ou abstração, todas originadas a partir de um mesmo objeto (EASTMAN, 1980b; 1991; MAHDAVI, 2003; IBRAHIM *et al.*, 2004; STOUFFS, 2008). Ibrahim e outros autores consideram que a modelagem paramétrica é recursiva – a representação resultante dos parâmetros iniciais definidos pelo usuário pode sugerir a necessidade de uma definição complementar. Ou então objetos são inseridos para que se obtenha uma representação preliminar, que posteriormente será aperfeiçoada. Por isso é necessário garantir diferentes comportamentos para o objeto, de acordo com o conjunto de parâmetros selecionados, do mínimo ao máximo de definições possíveis. Um objeto que tem a capacidade de manter-se funcional mesmo em contextos intermediários de definição de parâmetros denominado pelos autores como objeto semi-definido. (IBRAHIM *et al.*, 2003)

A grande quantidade de parâmetros que pode ser encapsulada em um objeto para simular adequadamente um elemento construtivo traz um grande potencial para a agregação de informações, mas é o comportamento dos objetos que viabiliza a extração de dados e oferece uma funcionalidade real para o processo de projeto. Se um projetista pretendesse utilizar um CAD baseado apenas em entidades geométricas sem comportamento (um CAD 3D como o 3D Studio, por exemplo) para produzir o

resultado mostrado na figura 3.03, ele enfrentaria as seguintes dificuldades: primeiro, teria que produzir manualmente uma visualização em planta, que é o resultado da interceptação de todos os elementos de um pavimento por um plano paralelo ao piso. Se houvesse elementos em outros pavimentos, eles teriam que ser ocultados (em *layers* ou de outro modo). Vários CADs 3D não oferecem a possibilidade de criar uma vista em corte, ou seja, uma instância do modelo, cortado por um plano definido pelo usuário. Nesse caso, o projetista teria que cortar de fato os elementos, criando segmentos de paredes a cada visualização que precisasse, e ocultando os que estivessem entre o ponto de observação e o plano de corte imaginário.

Mesmo que esse enorme e contraproducente desafio fosse aceito e concluído, um simples detalhe corroboraria a afirmação de Eastman sobre a impossibilidade de gerar representações simbólicas a partir de entidades geométricas simples: portas são convencionalmente representadas abertas em planta, e fechadas em elevação ou corte. Para atender a essa convenção, o projetista teria que rotacionar todos os elementos que constituem cada uma das folhas de todas as portas. Já que a atividade de projeto é iterativa por natureza, essa operação provavelmente teria que ser repetida em várias ocasiões. Tudo isso apenas produziria a geometria básica, resultante das operações de corte. Uma série ainda mais complicada de operações teria que ser realizada para adequar a representação às convenções de desenho técnico. Muitos CADs 3D sequer oferecem a opção de definir a espessura das linhas dos elementos, ou então possuem apenas duas possibilidades: linhas espessas para elementos cortados e estreitas para elementos em vista. Como se pode ver pela figura 3.03, a representação de edifícios requer um nível de especialização muito maior.

Por se valerem do comportamento dos objetos, os BIM CADs realizam toda essa complicada operação automaticamente, de modo transparente para o usuário. O comportamento permite codificar o conhecimento técnico a respeito dos elementos construtivos para que seja incorporado ao projeto. A geração de diferentes representações é o aspecto mais evidente desse conhecimento, mas também é possível programar comportamentos que funcionem como pré-processadores, organizando parâmetros e passando-os para aplicações de análise. Lee e outros autores demonstram a geração de comportamentos codificando uma série de objetos

paramétricos representando elementos de concreto pré-moldado. Entre os comportamentos estavam, por exemplo, a adequação automática das armaduras dos consolos dos pilares, de acordo com o tipo de encaixe da vigas ou da carga a ser suportada por elas. Um interessante sistema de notação de comportamentos desejados é proposto, para garantir que a transformação das intenções iniciais em algoritmos seja realizada corretamente (LEE *et al.*, 2006).

3.4 Modelo

O termo modelo refere-se a uma estrutura de informação que permita integrar diferentes fases do desenvolvimento. Desenhos bidimensionais, embora também possam ser considerados "modelos", na medida que representam abstratamente um objeto real, não transmitem informação suficiente para que se obtenha essa integração. Ibrahim e outros autores propõem a distinção entre os termos modelagem gráfica e modelagem de informação, para designar os modelos de acordo com o seu objetivo principal (IBRAHIM *et al.*, 2003). Modelos gráficos são construídos para permitir uma melhor visualização dos conceitos nas fases iniciais ou do resultado pretendido nas fases finais do desenvolvimento do produto, e em geral não exercem grande influência no processo de desenvolvimento como um todo. Um exemplo clássico são as maquetes eletrônicas, que são encomendadas por construtoras para auxiliar nas campanhas de marketing do produto, e pouco influenciam o processo de projeto (SPERLING, 2002). Modelos de informação, como os modelos BIM, seguem o paradigma da modelagem de produto: são principalmente veículos para a transferência eficiente de informação entre as fases do desenvolvimento.

Transferências de informação entre as fases de desenvolvimento dos edifícios sempre ocorreram, independente do veículo utilizado para a comunicação. Atualmente, o veículo mais utilizado são os desenhos criados em sistemas CAD, baseados em primitivos geométricos. O *software* Autocad é o exemplo mais notório desse tipo de CAD, e o seu formato ainda é considerado a plataforma mais largamente disseminada para produção de documentação na indústria da construção. Uma das principais críticas que se fazem à representação de edifícios por meio de desenhos, é que eles são formados por primitivos geométricos que são incapazes de informar

inequivocamente a natureza dos elementos construtivos que representam. Um exemplo clássico é a representação de paredes utilizando linhas paralelas: o CAD baseado em primitivos geométricos armazena pouca informação além da posição e da extensão das linhas que representam paredes. De fato, ele sequer é capaz de informar automaticamente se elas são paralelas, porque a definição do paralelismo é mantida apenas durante a criação das linhas. Por isso os diferentes elementos que são utilizados para representar paredes em CADs baseados em primitivos geométricos não possuem qualquer relação funcional entre si, cabendo ao usuário, treinado para reconhecer padrões convencionados, interpretá-los como paredes. A figura 3.04 ilustra essa situação: o conjunto de elementos que é interpretado pelo projetista como paredes, batentes e porta é formado por primitivos geométricos sem relacionamentos funcionais, e operações acidentais podem desfazer completamente o significado transmitido por eles.

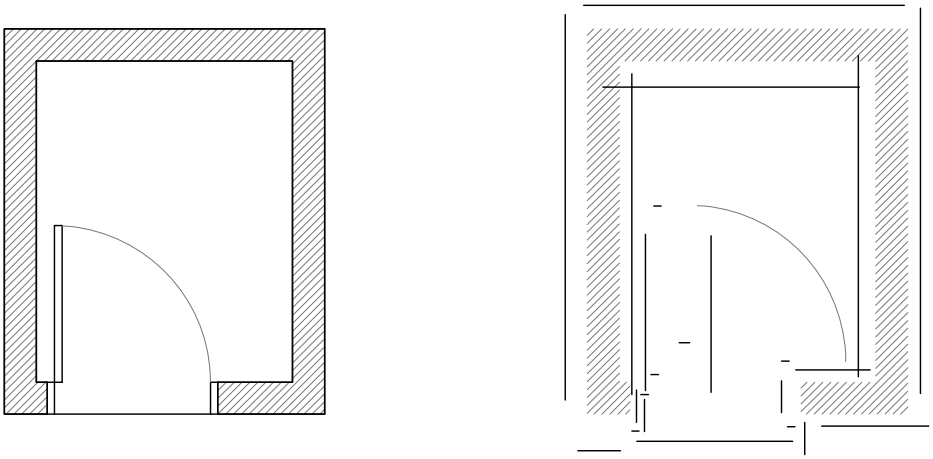


Fig. 3.04: representação por primitivos geométricos e a possibilidade de perda de significado após operações sobre os elementos individuais.

3.4.1 Semântica

A qualidade que garante a transmissão eficiente de informações entre diferentes fases do desenvolvimento de um edifício, impedindo a ocorrência de situações como esta, é a semântica. Essa é a maior distinção entre a representação de um edifício utilizando um modelo em relação à representação por desenhos. Os modelos BIM registram tanto os elementos construtivos como as relações funcionais entre eles, e também entre os elementos de representação que os descrevem

graficamente. Isso cria conjunto coerente que pode ser interpretado tanto por usuários como por computadores, e mantém o significado da informação durante as transmissões entre as fases de desenvolvimento (PENTTILA, 2005). Na definição da NBIMS, essa característica foi considerada essencial para a implementação da modelagem na indústria da construção (NIBS, 2007). A importância de garantir que usuários de diferentes disciplinas em várias fases do desenvolvimento do edifício compreendam a representação inequivocamente evita erros e reduz a necessidade de comunicações complementares para esclarecimento de dúvidas, aumentando a agilidade dos processos. A importância de garantir que os computadores compreendam a representação inequivocamente reduz a necessidade de tarefas de reentrada de dados e os possíveis erros humanos que podem surgir dessas operações.

Por exemplo, a representação de paredes em modelos BIM utiliza objetos especializados em representar paredes, com comportamentos e parâmetros definidos para garantir a coerência do resultado e a manutenção do seu significado como parede. Por isso, não seria possível desfazer completamente o significado da representação com operações similares às apresentadas na figura 3.04. A figura 3.05 ilustra a relação indissociável entre o objeto e os elementos gráficos que compõem a sua instância de representação em planta. A posição das paredes foi modificada, mas a natureza dos objetos foi mantida.

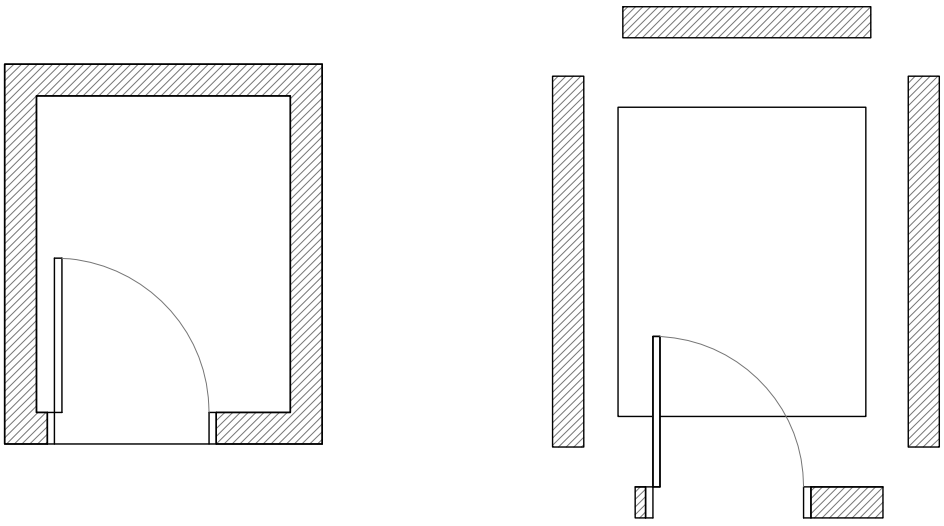


Fig. 3.05: a representação por objetos paramétricos em um BIM CAD mantém o significado da informação após operações de modificação.

Além de relações internas ao objeto – entre ele os elementos utilizados para a sua representação – podem ser estabelecidas relações entre diferentes objetos: todos os elementos de um pavimento são movidos caso a altura do pavimento inferior seja aumentada, por exemplo. Ou então um objeto representando um lance de escada pode ser subordinado a um determinado pavimento, e quando o pé-direito deste pavimento é modificado, o objeto recalcula o número e a altura dos degraus automaticamente. Essas funcionalidades foram descritas por Eastman há quase 30 anos, durante o desenvolvimento do sistema BDS (EASTMAN, 1980b; 1991).

Outra funcionalidade decorrente do relacionamento semântico entre os objetos é a possibilidade de se estabelecer regras ou condições a serem atendidas para que os objetos permaneçam coerentes. Isso permite, por exemplo, realizar análises para verificar conflitos espaciais ou mesmo funcionais a partir do modelo: identificando áreas internas e áreas externas, seria possível gerar uma regra que determinasse que todas as portas de entrada de um edifício público deveriam abrir para fora, e não para dentro (EASTMAN, 1991; 1992; LEE *et al.*, 2006). Também é possível estabelecer relacionamentos semânticos que atendam necessidades de fases específicas do desenvolvimento do edifício, como organizar os elementos por custo, data prevista para construção ou equipe designada para a atividade, por exemplo, o que é especialmente útil para a fase de planejamento da construção (EASTMAN *et al.*, 2004).

Do ponto do desenvolvimento de aplicações para acessar dados de modelos de edifícios, a estruturação semântica é a característica principal da tecnologia BIM. Sem ela, aplicações com essa finalidade seriam no máximo semi-automatizadas, ou seja, exigiriam do usuário operações de complementação da informação após a criação das representações, como informar ao sistema que tipos de elementos construtivos são representados por cada grupo de elementos geométricos em um arquivo. O trabalho de Wang e Messner, por exemplo, aponta essa como uma das dificuldades para a utilização de CADs 4D, que poderiam ser utilizados mais largamente para simular e controlar as fases da construção de edifícios, mas não são, em parte por exigem do usuário longas operações de transposição de informações (WAN e MESSNER, 2007).

3.5 Modelagem

Modelagem é o processo de inserção dos objetos paramétricos em um modelo do edifício. Aplicações com grande capacidade de modelagem de sólidos são essenciais nesse processo, uma vez que grande parte da manipulação da informação na construção se dá de forma gráfica (EASTMAN, 2004; LEE *et al.*, 2006). Essas aplicações de modelagem, centrais para o desenvolvimento dos modelos, também são chamadas BIM CADs ou *BIM-based* CAD (IBRAHIM *et al.*, 2004). O nome BIM CAD pode soar redundante, considerando que o propósito inicial da tecnologia CAD era fornecer um ambiente de projeto, não apenas de produção de desenhos (COONS, 1963). Contudo, a massiva adoção de *softwares* de desenho bidimensional baseados em primitivos gráficos durante as décadas de 1980 e 90 gradualmente incutiu na comunidade leiga uma interpretação incorreta do termo, e logo CAD passou a designar mais um segmento de *softwares* do que a utilização de todo o potencial do computador para auxiliar o processo de projeto. Nesse sentido, BIM é um retorno ao CAD como fora imaginado por seus pioneiros nas instalações do MIT na década de 1960. Porém, na atual fase de transição da representação por desenhos para a modelagem, o significado formado nas duas últimas décadas ainda provoca desentendimentos, motivo pelo qual a maioria dos autores citados neste trabalho preferiu qualificar o termo CAD, como CAD de modelagem, BIM CAD, CAD paramétrico, CAD integrado, etc.

3.5.1 Automatização da documentação

A geração automática da documentação projetual a partir dos objetos do modelo BIM é a vantagem mais imediata da utilização de um BIM CAD. Todos os principais sistemas comercialmente disponíveis tem rotinas para geração automática de várias formas de relatórios a partir do modelo, sejam eles gráficos ou textuais. Essa automação é possibilitada por rotinas especializadas que processam parâmetros dos objetos do modelo e formatam o resultado. Desenhos técnicos como plantas, cortes e elevações são considerados um tipo de relatório, descrito em forma gráfica, e por conseguinte são gerados automaticamente (EASTMAN, 1976; 1991; SACKS *et al.*, 2004). Como exemplos de relatórios textuais geralmente são citadas listas de elementos ou áreas e quantitativos de materiais. Rotinas também podem ser

programadas para realizar diferentes análises a partir dos parâmetros extraídos dos objetos (SACKS e BARAK, 2007).

BIM CADs produzem representações através do instanciamento da informação do modelo, e não de cópia dessa informação. Todos os relatórios gerados automaticamente, incluindo plantas, cortes, elevações e perspectivas, são arranjos temporários dos objetos do modelo, para produzir uma visualização adequada. Assim que o usuário determinar que não necessita mais da visualização, esses arranjos são desfeitos, sem prejuízo para a informação original. Essa abordagem traz uma grande vantagem para a modelagem BIM: a manutenção da consistência da informação. Uma vez que as diferentes visualizações são instâncias, modificações em seus elementos são de fato modificações no modelo do edifício. Quando o sistema detecta que o modelo foi modificado, atualiza a informação de todas as outras instâncias que estão ativas naquele momento (EASTMAN, 1991; IBRAHIM *et al.*, 2003; EASTMAN *et al.*, 2004). Por exemplo, quando se aumenta o vão de uma porta ou janela uma elevação, os parâmetros do objeto são modificados, e todas as outras instâncias que estiverem ativas, sejam elas gráficas ou textuais (como uma lista de esquadrias) passam a representar a porta com as novas dimensões.

É importante ressaltar, porém, que dificilmente o projetista será afastado completamente do processo de geração documentações e simulações. Por isso, pode-se pensar nessa funcionalidade mais como automação parcial da documentação, liberando o usuário das funções mais repetitivas, mas ainda dependendo dele para solução de situações mais complexas. Kalay afirmou que a figura do projetista permanece insubstituível, independentemente da evolução das tecnologias de informação, pois ele exerce sobre o conteúdo da informação uma ordenação semântica que dificilmente poderia ser transformada em algoritmos (KALAY, 1985b). Augenbroe considera também a influência preponderante da capacitação do próprio usuário na qualidade das informações produzidas em simulações dos edifícios, e a forte tendência para distribuição das simulações entre especialistas através da internet, em vez de inseri-las simplificadamente nos sistemas utilizados na documentação do projeto (AUGENBROE, 2002).

3.5.2 Organização da documentação

Representações instanciadas são arranjos temporários, mas é possível definir para elas estruturas permanentes de organização e relacionamento. Manusear e visualizar adequadamente as grandes quantidades de informação produzidas em projetos de edifícios pode tornar-se um problema mesmo com a utilização de modelos ao invés de desenhos. No início da tecnologia da modelagem, chegou a se imaginar que a distribuição da informação do projeto seria homogênea pelos espaços que o constituíam, com a representação do projeto sendo formada apenas por desenhos em grande escala. Essa alternativa mostrou-se inviável, porque a informação de um projeto não é distribuída uniformemente. A sua densidade varia de acordo com a necessidade de representação dos sistemas que constituem diferentes partes do edifício: juntas em estruturas de aço, por exemplo, contém uma considerável quantidade de conhecimento técnico agregado: como executar a montagem, quais peças são necessárias, em que sequência devem ser colocadas, que tolerâncias são permitidas e qual é o desempenho determinado pelas normas vigentes, e assim por diante. Por outro lado, a representação da extensão das vigas de aço possui muito menos conhecimento agregado, já que trata-se apenas uma extrusão contínua de uma seção padronizada. Eastman ponderou que a solução para essa variação de densidade de informação é reproduzir no computador o esquema de desenhos em diferentes escalas utilizado no projeto em papel. Desenhos em grande escala representam situações onde há muita instrução a ser transmitida. Desenhos em pequena representam a distribuição geral dos elementos e servem de índice para localizar os desenhos em grande escala (EASTMAN, 1980a). Nos BIM CADs, essa situação é resolvida com a definição de relações entre as diferentes instâncias de representação do modelo. Algumas relações são fixas, determinadas pela interface dos sistemas, como a representação do edifício por pavimentos. Também existem ferramentas que permitem ao usuário inserir uma indicação de relação a uma instância de representação sobre outra instância de representação, além de possibilitar a configuração das suas propriedades (código, escala, nome, etc). Essa operação cria arranjos permanentes que são registrados no modelo. Por exemplo, existem ferramentas para dispor e configurar cortes, que aparecem em planta como uma

simbologia, de acordo com as convenções de desenho técnico adotadas. O mesmo pode ser feito para inserir indicações de detalhes nas plantas ou nos cortes. Dessa maneira, instâncias de representação mais abrangentes, como a planta, servem de índice para organizar instâncias mais específicas, como cortes e detalhes. Em geral, as próprias simbologias da indicação das instâncias mais específicas possuem funcionalidades que permitem ao usuário acessá-la.

Outro importante aspecto nesse sentido foi apontado por Eastman no início da década de 1990: um modelo de edifício deveria ser idealmente composto por dois conjuntos: um grupo de objetos paramétricos e outro de pranchas de desenho. Como as várias instâncias de representação dos objetos são arranjos temporários, deveria ser fornecido ao usuário um modo de fixar grupos de representações, compondo então pranchas de desenho técnico para visualização ou impressão (EASTMAN, 1991). Os BIM CADs atuais oferecem essa funcionalidade, registrando e mantendo as diferentes configurações definidas pelo usuário para pranchas (ou *lay-outs*). Pranchas podem conter várias vistas do modelo, com parâmetros de representação diferentes. Como são instâncias permanentemente ativas, todas as modificações no modelo atualizam-nas automaticamente.

3.5.3 Consistência da informação

O grande paradigma a ser substituído na adoção da tecnologia de modelagem de edifícios é o projeto centrado na criação de desenhos pelo centrado em um modelo do edifício (HIETANEN e DROGEMULLER, 2008). A vantagem da instanciação de representações nessa substituição fica evidente quando se considera o seu aspecto mais importante: a garantia da consistência da informação. No processo de projeto baseado em desenhos, essa consistência é obtida manualmente, em tarefas complexas e extremamente sujeitas a erros, característica que foi observada já no início da tecnologia de modelagem de edifícios (EASTMAN, 1980a). O manuseio de diferentes porções de informação do projeto no Autocad é um exemplo típico desta situação: não há mecanismos intuitivos para trabalhar com as várias plantas que compõem a representação de um edifício de múltiplos andares. As diferentes plantas dos pavimentos podem ser separadas por arquivo, por *layer*, ou então dispondo os

elementos de cada planta em regiões diferentes do mesmo arquivo, como se fossem representadas em uma enorme folha de desenho. Ao representar o edifício através de desenhos, CADs baseados em primitivos geométricos forçam o projetista a transpor manualmente informações entre as diferentes vistas isoladas. Essa transposição é feita apenas visualmente, comparando os diferentes desenhos, ou então copiando um conjunto de elementos (por vezes a vista inteira) que serve de base para a propagação manual da modificação para as demais vistas. Dada a natureza iterativa do projeto de edifícios, essas transposições ocorrem várias vezes, em ambos os sentidos, sempre que interferências são percebidas ou modificações se fazem necessárias (AYRES e SCHEER, 2007).

Várias formas de complementar o conteúdo da informação e facilitar o seu manuseio e a transmissão foram desenvolvidas. A definição de padrões para *layers* é um exemplo dessa tentativa de semantizar o conteúdo desestruturado dos CADs baseados em primitivos geométricos (HOWARD e BJÖRK, 2007). No Brasil, a AsBEA, associação dos escritórios de arquitetura, propôs a padronização da nomenclatura dos *layers* e da estrutura de pastas onde os arquivos de diferentes projetos complementares deveriam ser armazenados (CAMBIAGHI, 2000). Para outras situações, são desenvolvidas metodologias e convenções próprias em cada escritório, na tentativa de compensar a estrutura de informações deficiente dos CADs baseados em primitivos geométricos. Por mais que possam agilizar o processo e ampliar o conteúdo de informações dos desenhos, essas adaptações são seriamente limitadas pelo modelo de dados simplificado utilizado pelos CADs baseados em primitivos geométricos. O registro da informação adicional é frágil, e pode ser facilmente corrompido.

Chastain e outros autores fazem interessante crítica sobre o uso de computadores na fase de projeto de edifícios, afirmando que certas implementações dessa tecnologia não consideraram adequadamente o seu contexto, e não se ajustaram perfeitamente às demandas, como um “bloco quadrado sendo forçado por um buraco redondo”. Em outros casos, implementações do computador na produção de projetos incorreram no erro cuja analogia é a “carruagem sem cavalos” (o nome pelo qual os primeiros automóveis foram vulgarmente chamados). Ou seja, o potencial

da tecnologia não foi plenamente compreendido, e tanto a sua explicação como a implantação foram realizadas a partir de um esquema conceitual desenvolvido para as tecnologias anteriores, que a nova tecnologia deveria substituir (CHASTAIN *et al.*, 2002). Considerando o potencial de organização de informações oferecido pelo computador, produzir projetos em CADs usando exatamente a mesma metodologia que era utilizada com desenhos em papel realmente justifica o termo "prancheta eletrônica", que embora soe futurista deve ser interpretado como uma desvantagem para o processo de projeto (AYRES e SCHEER, 2007). Ao automatizar completamente a manutenção da consistência da informação do modelo, a BIM aumenta o desempenho do projetista e reduz a possibilidade de erros, utilizando o potencial do computador de maneira mais adequada (PENTTILA, 2005).

3.5.4 Generalização e extensibilidade

Vários trabalhos observam que sistemas CAD bem sucedidos permitem a representação de inúmeras situações de projeto, como diferentes sistemas construtivos ou níveis de detalhamento. Eastman, por exemplo, atribui o sucesso dos CADs baseados em primitivos geométricos a essa característica de generalização. No outro extremo estão sistemas CAD especializados em sistemas construtivos fechados, para os quais a informação completa sobre os elementos já está disponível no início do processo de projeto. Estes sistemas, por mais eficientes que possam ser em seus domínios restritos, nunca se adaptarão à condição de incerteza que normalmente envolve as construções típicas, que utilizam sistemas construtivos abertos (EASTMAN, 1976; 1989; 1992; 2004). Em sistemas de modelagem BIM, um conjunto de objetos atômicos (ou "nativos") é utilizado para representar os principais elementos construtivos de maneira genérica. Objetos representando paredes, por exemplo, podem ser descritos como prismas simples, sem maiores especificações. Esta representação é suficiente para descrever simplificadamente paredes de qualquer sistema construtivo, porque se concentra na característica comum a todas as paredes, que é funcionar como um painel separando espaços.

Mesmo adotando essa abordagem genérica, não é possível criar um sistema CAD que atenda a todas as situações que podem surgir no desenvolvimento de

projetos. Diferentes tipos de construções, realizadas por diferentes empresas sob os mais variados códigos de construção devem ser modeladas (SACKS *et al.*, 2004). Por isso, os BIM CADs oferecem a possibilidade de extensão através da criação de *scripts* ou aplicações que complementam as suas funcionalidades. Eastman classificou essa qualidade como fundamental para para os sistemas de modelagem de edifícios (EASTMAN, 1976; 1980b; 1989; 1992). Outra forma de extensão é a possibilidade de desenvolvimento de novos objetos paramétricos pelo usuário, a partir de objetos paramétricos existentes. Desse modo, funções genéricas podem servir de base para a criação de ferramentas que atendam a necessidades de situações ou sistemas construtivos específicos (EASTMAN, 1991).

3.6 Metamodelagem

A construção de edifícios sempre foi uma atividade conjunta que dependeu da cooperação de vários profissionais com diferentes conhecimentos técnicos. Mesmo se consideradas as suas diferentes fases isoladamente, a necessidade de trabalho cooperado continua. Dificilmente se imagina o projeto arquitetônico sendo realizado individualmente, por exemplo (KVAN, 2000). As diferentes disciplinas envolvidas no desenvolvimento de edifícios produzem soluções que são completas apenas se for considerado o conjunto de técnicas da disciplina isoladamente. Considerando a informação necessária para a realização da totalidade do processo, essas soluções são parciais e precisam ser integradas. No momento da integração podem surgir conflitos entre as soluções das diferentes disciplinas, e gerenciar a resolução destes conflitos torna-se cada vez mais difícil, porque porque o conhecimento técnico é cada vez mais complexo e ao mesmo tempo segmentado em diferentes profissões (KALAY, 2005).

Até hoje, esse tipo de integração de informações de diferentes domínios não tem sido prático, pois as ferramentas de projeto não fornecem funcionalidades adequadas para a visualização de grandes quantidades de informação ou então exigem extensos processos manuais de reentrada de dados e verificação de conflitos (MAHDAVI, 2003). Idealmente, a informação agregada por diferentes disciplinas deveria fluir sem obstáculos entre o projeto, a fabricação, a construção, a manutenção, e todas as outras atividades interrelacionadas que constituem o desenvolvimento de

um edifício, ficando disponível a todos os envolvidos automaticamente. Este é o principal objetivo da BIM, que permite que o edifício seja construído virtualmente, dentro do computador, antes da construção real no canteiro. Essa abordagem reúne todos os envolvidos em um arranjo virtual de projeto cooperativo, permitindo que o conhecimento agregado por cada profissional seja integrado em uma única representação – o modelo – que é disponível a todos os outros profissionais participantes. Um único modelo integrado do edifício permite relacionar melhor as informações e facilita a manutenção da consistência e da integridade dos dados (EASTMAN, 2004). Desse modo, é possível verificar e solucionar antecipadamente os conflitos entre definições de diferentes disciplinas, reduzindo a ocorrência de erros e aumentando a qualidade do produto.

Desenhos, por melhor elaborados e convencionados que sejam, cobrem apenas fases isoladas e instantâneos do processo de desenvolvimento do edifício (KALAY, 1985b; PENTTILA, 2005). Um modelo de edifício, por outro lado, é multidimensional por natureza, abrangendo toda a informação produzida e utilizada durante todas as atividades do ciclo de vida do edifício, em um processo gradual de agregação de conhecimento (SACKS *et al.*, 2004; TSE *et al.*, 2005). Da fase de projetos, o modelo passa para todas as subsequentes, como o planejamento da construção e a sua administração, fabricação de componentes, comissionamento, operação da edificação e as suas manutenções periódicas (EASTMAN, 1992). A questão de definir se esse fluxo de informações seria melhor obtido pelo uso de um único modelo integrado, ou de vários modelos interconectados é mais uma decisão contratual do projeto, e não tem grandes impactos sobre o princípio da BIM. Seja a informação reunida fisicamente em um único lugar ou distribuída de acordo com as responsabilidades individuais, a BIM é baseada na possibilidade de coletá-la e integrá-la automaticamente (EASTMAN *et al.*, 2004).

3.6.1 Interoperabilidade

BIM é um conceito amplo que não pode ser utilizado para descrever um tipo de *software*. Esse seria o mesmo erro cometido durante a disseminação do conceito CAD, que ficou mais relacionado às aplicações de desenho bidimensional do que ao

processo de projeto auxiliado pelo computador. Tampouco pode-se contemplar a sua totalidade pela utilização de um único software, porque não há aplicações que abranjam todo o ciclo de vida de um edifício. Sistemas dessa natureza seriam complexos e rígidos demais para serem úteis ao processo de modelagem. Ao contrário, o desenvolvimento para a BIM deve continuar orientado para a criação de aplicações específicas para as várias disciplinas envolvidas na construção (EASTMAN *et al.*, 2004).

A criação de modelos BIM se dá em um sistema composto por vários tipos de aplicações, com diferentes objetivos e enfatizando diferentes porções da informação, colaborando e compartilhando dados (IBRAHIM *et al.*, 2004). A troca de informações entre essas várias aplicações deve ocorrer sem sobressaltos, garantindo que o significado não seja prejudicado. O termo que define esse requisito é "interoperabilidade", que pode ser entendida como um mapeamento das estruturas internas de dados das aplicações envolvidas em relação a um modelo universal, independente de fabricantes (ou neutro). Com isso, novas aplicações podem ser desenvolvidas partindo desse mapeamento, eliminando a prática onerosa de criar várias rotinas de transposição de informação para cada aplicação ou versão utilizada no desenvolvimento de edifícios (NIBS, 2007).

Para Brunnermeier e Martin, interoperabilidade é a habilidade de comunicar os dados do produto através de diferentes atividades da produção. A interoperabilidade é essencial para a produtividade e a competitividade de muitas indústrias, porque projetos e processos de fabricação eficientes requerem a coordenação de muitos participantes e processos diferentes que dependem de representações digitais do produto. Em seu relatório sobre a inadequação dos processos de troca de informações entre as fases do desenvolvimento de automóveis na indústria americana, eles estimaram que o custo da interoperabilidade inadequada, causado por erros, atrasos e retrabalhos, é de no mínimo um bilhão de dólares por ano (BRUNNERMEIER e MARTIN, 1999). Porém, a situação na indústria da construção é ainda mais crítica. Gallaher e outros autores prepararam relatório semelhante para a indústria da construção americana, considerando apenas as obras de infra-estrutura, e estimaram um custo de 15.8 bilhões de dólares por ano (GALLAHER *et al.*, 2004).

Existem ainda vários desafios relacionados à interoperabilidade a serem superados. Entre são decorrentes das atuais práticas projetuais (contratação, responsabilidades, entregas) e responsabilidades legais. A modelagem de produto é uma ruptura significativa nos métodos tradicionais de projeto e vai exigir a adoção de muitas novas abordagens e procedimentos. A contratação de projetos, por exemplo, pode passar a se basear no nível de aprimoramento do modelo, e não na quantidade ou detalhamento dos desenhos (EASTMAN, 1991). Um outro desafio preocupante é a questão da proteção da informação inserida no modelo, que pode ser acessada por todos os outros participantes, o que dilui a noção de responsabilidade técnica. A organização em torno da BIM vai exigir formas seguras para controlar o uso da informação contida no modelo. Por exemplo, ferramentas precisarão ser desenvolvidas para atribuir níveis de acesso a indivíduos, grupos ou arranjos temporários interdisciplinares; para permitir que se trace a origem, as modificações graduais e as diferentes versões resultantes da informação; e também para gerenciar e garantir a origem da informação da documentação legal gerada a partir do modelo, como projetos para aprovação, listas de materiais para contratação de serviços, ou mesmo quantitativos para licitações (NIBS, 2007).

Outro grupo de desafios é relacionado às capacidades atuais das ferramentas de modelagem e das estruturas de dados disponíveis. Ainda não ficou suficientemente claro, por exemplo, como será garantido o acesso simultâneo ao modelo. Apenas pequenas edificações podem ser projetadas por um único indivíduo em um período de tempo praticável. A modelagem de edifícios maiores exige o acesso simultâneo de usuários a grandes porções de informação, para permitir a atualização constante. Novas formas de controle são necessárias para permitir que equipes formadas por um grande número de projetistas trabalhem eficientemente (EASTMAN *et al.*, 2004). Além disso, propriedades globais do projeto, como objetivos gerais, prazos, organização dos participantes, devem ser representadas em um nível acessível a todos os envolvidos e relacionarem-se com informações específicas para cada disciplina ou fase – por exemplo, os objetivos específicos para o projeto das instalações hidráulicas (EASTMAN, 1992).

3.6.2 IFC - *Industry Foundation Classes*

A idéia de integrar dados de diferentes aplicações especializadas entre as etapas do desenvolvimento de edifícios permeou toda a história do uso do computador na construção civil. Gingerich, em 1973, comenta a existência de diversos programas de computador para auxiliar o desenho na arquitetura. Eles variavam de algoritmos de alocação de espaços a análises estruturais e computação gráfica. Geralmente, estes programas eram independentes, e mesmo que muitos deles utilizassem estruturas de dados similares para representar edifícios, suas partes e sistemas, eles raramente possuíam formatos compatíveis. Consequentemente, para executar um determinado programa, o usuário precisava redescrever o edifício e suas partes. Como possível solução, o autor sugeriu que empresas, faculdades ou – preferencialmente – entidades de normatização especificassem um formato de banco de dados para definição de edifícios, a ser utilizado por todos os programas. Isso economizaria o tempo e o trabalho gastos na transformação manual de dados entre aplicações. Tal banco de dados deveria ser flexível o suficiente para lidar com os processos de aquisição, apresentação e modificação, e também suficientemente inclusivo para que fosse possível definir toda e qualquer parte do edifício até o nível necessário para a fase de construção (GINGERICH, 1973). Na indústria da manufatura, a idéia de padronizar modelos de dados resultou na STEP. Porém inicialmente nenhum dos seus *Application Protocols* (APs) atendia aos requisitos especiais da indústria da construção.

Em agosto de 1994, doze companhias americanas reunidas em torno da Autodesk juntaram esforços para verificar a possibilidade de fazer diferentes aplicações trabalharem de modo integrado, com base no então recente Autocad versão 13. O grupo de trabalho foi chamado inicialmente *Industry Alliance for Interoperability*, e concluiu que a interoperabilidade poderia trazer benefícios econômicos significativos, desde que pudesse ser demonstrada na prática. Meses depois, os primeiros resultados foram mostrados e despertaram grande interesse da indústria. Várias empresas desenvolvedoras de *software* insistiram em participar da iniciativa, e as inscrições para membros foram abertas em 1995. O principal objetivo do grupo passou a ser o desenvolvimento de padrões independentes de fabricantes

para interoperabilidade dos *softwares* utilizados na indústria da construção, utilizando a linguagem de definição de dados EXPRESS, que havia sido lançada no ano anterior, na norma STEP. Em 1997, a organização foi reconfigurada em uma entidade sem fins lucrativos, com o objetivo principal de desenvolver um modelo de dados neutro para representar o ciclo de vida dos edifícios, e o seu nome foi trocado para *International Alliance for Interoperability*, a IAI (EASTMAN, 1999; IAI, 2008c). O modelo de dados neutro da IAI é chamado *Industry Foundation Classes*, ou IFC. A sua primeira versão foi lançada em 1997, e a versão atual é a 2X3, sendo que a 2X4 *alpha* já está disponível para testes na página da IAI na internet (IAI, 2008c). As IFC são atualmente o programa de padronização de modelos de edifícios mais ambicioso da indústria (HOWARD e BJÖRK, 2008).

O desenvolvimento das IFC aborda a massiva quantidade de dados que podem ser inseridas em um modelo de edifício em quatro eixos de informação: ciclo de vida, disciplina, nível de detalhe e aplicações (*softwares*). A proposta não é criar uma representação específica para cada elemento encontrado na construção, já que isso daria origem a um modelo muito grande e pouco implementável. Ao invés disso, os elementos são representados por classes genéricas, com informação suficiente para descrever as suas características principais. Também existe a possibilidade de estender uma descrição para que a representação se adéque melhor a um produto específico. A arquitetura do modelo de dados IFC é composta por quatro níveis: *domain*, *interoperability*, *core* e *resource*. O nível *domain* é o mais alto, e permite a descrição de informações específicas para cada disciplina envolvida no desenvolvimento do edifício. No nível *interoperability* são descritos mapeamentos de dados para permitir a troca de informação entre diferentes *domains*. No nível *core* são descritas as unidades de informação comuns a todos os domínios e mapeamentos, que podem ser especializadas por eles. O nível mais baixo é o *resource*, que contém a descrição de conceitos básicos e independentes, que são utilizados pelos níveis mais altos. A primeira versão das IFC tinha cinco tipos de *resource*: *geometry*, *property*, *property type*, *measure* e *utility*. Na versão 2X essas unidades foram subdivididas e outras foram acrescentadas, resultando em 20 tipos de recursos (LIEBICH e WIX, 2000).

Como é desenvolvido em EXPRESS, a sintaxe do modelo de dados (ou esquema) IFC utiliza os constructos mostrados na subseção 2.3.4. As unidades básicas de informação são entidades, que podem representar objetos, propriedades dos objetos, ou relações entre objetos (*ifcObject*, *ifcPropertyDefinition*, *ifcRelationship*). Objetos podem descrever elementos físicos, como paredes, espaços, equipamentos, mas também abstratos como tarefas, controles, processos, etc. Eles são derivados em sete tipos principais: produtos, processos, controles, recursos, atores, projetos e grupos (LIEBICH e WIX, 2000). As extensões das entidades básicas do modelo são criadas com a propriedade EXPRESS "*SUBTYPE OF*", e em teoria, desde que enquadrem-se na arquitetura da IFC, podem ser interpretadas inequivocamente em qualquer aplicação que consiga ler o modelo. O esquema IFC completo pode ser obtido no site da IAI (LIEBICH *et al.*, 2006).

Para que um fluxo contínuo de informações realmente possa ocorrer, três fatores devem ser atendidos: o formato no qual a informação é trocada, um entendimento comum a respeito do significado da informação sendo trocada, e a definição de qual informação trocar e quando realizar a troca. Na visão da IAI sobre a interoperabilidade, estes três requisitos são contemplados pelo modelo IFC, que é responsável pelo armazenamento digital, pelo IDM (*Information Delivery Manual*), que dá suporte aos processos do desenvolvimento de construções, e pelo IFD (*International Framework for Libraries*), que define a terminologia dos elementos do projeto. Para expandir a utilização do modelo IFC e melhorar a comunicação entre as etapas da construção, o desenvolvimento de *softwares* deve atender aos três requisitos (KIVINIEMI *et al.*, 2008).

IFD

IFD determina a terminologia dos elementos do projeto de construções e a relaciona com as entidades IFC. Além do nome pelo qual os elementos que compõem os edifícios são chamados variar entre diferentes regiões, as partes que compõem cada elemento também variam. Um exemplo disso são as portas: pode-se considerar que o elemento porta seja uma unidade completa, com folha, caixilhos e ferragens. Ou então apenas a folha e os batentes, ou então o caixilho pode ser um quadro completo,

dispensando a instalação da soleira, ou somente um arco, e assim por diante. Como as IFC são flexíveis e extensíveis, este elemento pode ser descrito de várias maneiras. Um entendimento comum sobre os elementos descritos nos modelos IFC era necessário, ou no mínimo um esquema que permitisse mapear as diferenças entre as descrições regionais dos elementos. Esta é a proposta do *International Framework for Dictionaries*, ou IFD, que pretende estabelecer dicionários ou ontologias para definir semanticamente a natureza dos objetos que descrevem elementos construtivos (IAI, 2008b).

IDM

IDMs especificam qual informação deve ser trocada em cada cenário possível durante as atividades do desenvolvimento do edifício, e relaciona essa definição com as entidades IFC. Por exemplo, qual é o conjunto necessário de informações a ser transmitido do projeto arquitetônico para o projeto de instalações elétricas para que o trabalho possa ser realizado. As definições dos IDMs são implementadas através de esquemas MVD (*Model View Definition*), descritos na forma de texto ASCII, de forma a serem lidos tanto por computadores como usuários. A definição IDM em conjunto com o esquema resultante, MVD, realiza sobre o modelo IFC uma espécie de ordenamento e seleção das entidades, provendo às diferentes disciplinas envolvidas na construção uma visão própria sobre os dados (IAI, 2008b).

Desafios para o modelo IFC

Dez anos após o início do desenvolvimento das IFC, o seu uso ainda se limitava a projetos piloto e testes de conformidade. Kiviniemi fez importante relato sobre a situação da tecnologia e da IAI à época. O autor comenta inicialmente a falta de recursos, e o reduzido número de poucas pessoas realmente envolvidas com o desenvolvimento das IFC. No aspecto da interoperabilidade, o processo de certificação de *softwares*, ainda muito simplificado, não garante a qualidade dos mapeamentos de dados durante a importação ou exportação entre diferentes aplicações. A falta de documentação e de mecanismos mais eficientes para atualização das informações para os interessados também é mencionada. Para Kiviniemi, a questão técnica da definição do esquema avançou consideravelmente, já existem *softwares* robustos para a

modelagem de edifícios, e não há mais razões para adiar a sua adoção, mesmo que a interoperabilidade seja limitada no início. A questão da adoção pelo mercado já provou ter solução a partir de iniciativas de grandes clientes ou entidades governamentais, e as demandas atualmente são de desenvolvimento de ferramentas e conscientização dos envolvidos (KIVINIEMI, 2006).

Trocas de dados sem perda de significado entre diversas aplicações ainda não são praticáveis, em parte porque os processos de mapeamento de dados entre os modelos internos dos BIM CADs e o modelo IFC ainda são imprecisos. Pazlar e Turk realizaram uma sequência de testes de exportação e importação de modelos de edifícios para o formato IFC utilizando três das aplicações mais conhecidas: Architectural Desktop 2005, Allplan Architecture 2005 e ArchiCAD 9. Os testes permitiram observar sérias inconsistências na representação dos elementos geométricos após o processo de exportação/importação, principalmente com modelos mais complexos. Os autores concluem que o ideal da interoperabilidade, embora difundido a mais de dez anos, ainda está bastante distante da aplicação na prática (PAZLAR e TURK, 2008).

Além disso, como exposto na subseção 3.3, é importante para a construção que se represente o edifício como um conjunto de objetos tridimensionais, porém as representações bidimensionais, essencialmente simbólicas, continuam indispensáveis. O ideal seria obter essa representação automaticamente, valendo-se do comportamento dos objetos (ver subseção 3.3.2). Porém a versão atual do esquema IFC não suporta essa visão, e quando o modelo do BIM CAD é exportado, toda a informação sobre a representação simbólica é perdida. O que passa a constituir o modelo IFC é um modelo de sólidos geométricos melhorado, com relações hierárquicas e descritores não geométricos, porém sem o comportamento de contexto que representa grande parte da sua utilidade para o projeto. Kim e Seo alertam inclusive para o fato de ser comum usuários inserirem elementos bidimensionais sobre os elementos tridimensionais nos BIM CADs, para que a representação em planta seja mantida durante a exportação para o formato IFC. Em recente trabalho eles apresentam o andamento do projeto XM-4, liderado pelo capítulo coreano da IAI, que pretende estender o modelo para comportar essa capacidade de extração de

documentação automaticamente. Além da topologia e da geometria, será necessário incluir como atributos das entidades: definição de *layers*, de linhas (espessura, tipo de traço), anotações, representações simbólicas, hachuras e padrões de preenchimento e curvas de Bezier. Outro projeto de extensão do modelo IFC é o XM-9, que se concentra em incluir no modelo a visão de pranchas, vistas, cotas associativas e capacidades para representações mais complexas (KIM e SEO, 2008).

Aranda-Mena e Wakefield afirmam que muito do insucesso da disseminação das IFC está relacionado com elas terem sido formuladas sobre conceitos que já não são universalmente aceitos como as melhores soluções para os problemas originais. Um exemplo é a questão da informação centralizada: para os autores, as pesquisas já apontam para uma situação de vários modelos auto-organizáveis, conectados entre si. Outro exemplo é a limitação da EXPRESS com relação à representação de relacionamentos semânticos: as ontologias são um campo emergente que permitem descrever a informação em um nível semântico muito maior. Apesar disso, as IFC obtiveram relativo sucesso, e os futuros desenvolvimentos de formatos de dados para interoperabilidade na indústria da construção provavelmente não desconsiderarão isso. Ao contrário, eles deverão ser criados a partir das IFC, adaptando as idéias iniciais às novas possibilidades das tecnologias de informação (ARANDA-MENA e WAKEFIELD, 2006).

3.7 Perspectivas para a BIM

As potencialidades e as possíveis causas para sua pequena adoção na prática são temas dos estudos sobre modelagem de produto na indústria da construção há mais de trinta anos. A resistência à mudanças e as condições especiais da indústria da construção foram constantes barreiras a uma adoção mais generalizada da tecnologia, a despeito das suas vantagens, que vem sendo anunciadas desde o final da década de 1970. Muito dos temas pesquisados atualmente ainda poderiam ser explicados por artigos de décadas atrás.

Eastman, por exemplo, afirmou que a relutância da indústria em adotar a modelagem de produto teve como causas a falta de pesquisa de modelos semânticos e abertos, que melhor representassem o edifício e os seus processos de construção e

permitissem a troca de dados entre as aplicações; a falta de ferramentas de modelagem mais intuitivas que se aproximassem do modo de trabalho dos projetistas; os benefícios potenciais limitados causados pela fragmentação da indústria; e as disputas de responsabilidades legais entre as diferentes disciplinas envolvidas nos projetos (EASTMAN, 1989). Uma pesquisa realizada em 2005 junto a empresas de construção de Hong Kong a respeito das principais ferramentas de projeto revelou que entre 788 projetistas entrevistados, 93% afirmaram utilizar o Autocad, 85% afirmaram utilizar o 3DStudio Max ou o 3DStudio VIZ e 29% o MicroStation (TSE *et al.*, 2005). Quase um terço dos respondentes já havia testado e adotado a modelagem de produto, mas o processo de construção da grande maioria dos edifícios em um grande centro urbano como Hong Kong provavelmente depende da frágil operação de reconstrução da informação a partir de uma grande quantidade de desenhos bidimensionais isolados. Além disso, dois softwares de modelagem gráfica e animação que não oferecem funcionalidades coerentes com as operações realizadas em um projeto de edificação são intensamente aplicados.

Também é recorrente a questão da adaptação das ferramentas BIM às fases menos "modeláveis" do processo de projeto, como a concepção arquitetônica ou de estruturas. Nesses casos, a possibilidade de modelar o edifício e analisá-lo não garante necessariamente dar suporte ao desenvolvimento de projetos, que começa com esboços, passa por desenhos esquemáticos e termina com a produção da documentação. Fases iniciais do processo de projeto não consistem em posicionar vigas, canos e outras peças de catálogo. Consistem em definir e compor abstrações variadas, que podem ser espaços, estações de trabalho, massas do edifício, superfícies visuais, entre outras. A dependência por abstrações nas fases iniciais do projeto é uma prerrogativa do projetista no desenho manual, e também deveriam ser do desenhista no computador. Os primeiros sistemas CAD que pretendiam oferecer suporte a mais de um tipo de atividade de projeto, porém, normalmente fixavam a sequência de operações do projetista. A razão para isso é que esquemas de verificação da integridade do projeto só funcionavam se a informação fosse inserida em uma determinada ordem. Sistemas CAD deveriam deveriam suportar um vasto repertório de abstrações e várias possibilidades de sequenciamento do desenvolvimento do

projeto. O projetista então poderia escolher o modo de trabalho que lhe parecesse mais adequado. Poucas pessoas que seriamente consideram-se "projetistas" deveriam aceitar uma organização do projeto (manual ou no computador) que pré-especifica rigidamente as abstrações a serem usadas e a sua sequência de aplicação (EASTMAN e HENRION, 1979; EASTMAN, 1980a). Atualmente, essa continua sendo uma situação mal resolvida. Turk, por exemplo, afirma que a tentativa de criar “modelos de concepção”, que definem a natureza dos objetos e relações que representam um edifício carece de estudos mais aprofundados, e tem comumente sido abordada a partir da questão técnica apenas. A simples proposição de gerar projetos de edifícios a partir de modelos parece para ele contraditória, pois a concepção de idéias orientada por modelos, sejam rígidos ou não, pode impedir o projetista de vislumbrar soluções originais que não se adéquem à estrutura do modelo (TURK, 2001). Ibrahim e outros autores fazem afirmação semelhante referindo-se às capacidades oferecidas pelos objetos paramétricos (IBRAHIM *et al.*, 2003).

Com relação à integração dos processos de projeto e simulação, Augenbroe afirma que a disponibilidade de ferramentas por si só não significa que a atividade exercerá influência sobre a evolução do projeto de edifícios. Para isso, é preciso garantir que a simulação seja realizada na hora certa, e pelos motivos certos. Isso requer tanto uma garantia de qualidade da simulação quanto uma coordenação mais adequada do processo de projeto. Além disso, modelos de dados para a integração “fácil” dessas ferramentas com os sistemas CAD são uma promessa há décadas, e houve poucos resultados práticos (AUGENBROE, 2002).

Chastain e outros autores afirmam que a subestimação dos potenciais das novas tecnologias de informação sobre o modo de conduzir as atividades da construção resulta na subutilização do seu potencial revolucionário e em aplicações inadequadas aos seus contextos. Para eles, tecnologias como a modelagem de produto só podem ser plenamente realizadas com a reorganização de grande parte dos processos, que ainda são orientados por um paradigma originado em um contexto anterior à informática (CHASTAIN *et al.*, 2002). Para Kalay, é preciso transformar a atual estrutura hierárquica e sequencial do processo de projetos para uma estrutura de atividades paralelas e mais efetivamente relacionadas (o que chamou de

interleaved process). A informação, nesse novo processo, seria um recurso plenamente acessível a todos os participantes, instantaneamente (KALAY, 2005).

Mahdavi sugere quatro abordagens para compreender a atual utilização dos modelos de edifícios e as novas possibilidades e desafios para a nova geração de aplicações. A primeira abordagem tem recebido mais atenção da comunidade científica: a integração da representação, com a utilização de repositórios únicos ou interconectados para reunir e organizar toda a informação sobre o edifício. Modelos, porém, podem oferecer funcionalidades muito mais sofisticadas, como a inversão do modo tradicional de inferência sobre os elementos. Habitualmente, pensa-se no processo de modelagem como definição de objetos paramétricos dos quais posteriormente se extraem dados para análises diversas. Os resultados das análises são utilizados então para redefinir estes objetos. Mahdavi propõe como segunda abordagem o projeto orientado pelo desempenho, pelo qual a definição prévia de requisitos orienta automaticamente a inserção dos objetos paramétricos configurados para atender ao desempenho desejado para a edificação. Uma terceira abordagem levaria a modelagem um passo adiante: a consideração do desempenho não de um campo de simulação, mas sim de todos os que o projetista considerar necessários. Sistemas de modelagem com essa funcionalidade teriam que combinar as diferentes definições para o desempenho da edificação provenientes das suas respectivas simulações e identificar o melhor conjunto de atributos para os objetos paramétricos. Por fim, Mahdavi propõe uma quarta abordagem, que é a extensão do modelo resultante das simulações de performance para as fases de operação e manutenção do edifício (MAHDAVI, 2003).

Apesar da necessidade de aumentar a quantidade e a profundidade dos estudos sobre a modelagem, a aceitação das suas vantagens para a indústria da construção cresce paulatinamente. O instituto de pesquisas VTT, da Finlândia, publicou recentemente a versão atualizada do *Finnish ICT Barometer*, uma pesquisa realizada através de e-mails para empresas do setor naquele país. Foram contabilizadas 86 respostas válidas, incluindo todas as especialidades, de arquitetos e engenheiros a proprietários. 93% das empresas afirmaram utilizar BIM em seus projetos. Porém, 75% dos projetistas afirmaram que a maior parte das documentações é criada em CADs

bidimensionais, e 81% dos profissionais não utilizam o modelo BIM compartilhado. Além disso, a maioria dos projetistas utiliza apenas o aspecto geométrico da modelagem, sem que a informação flua para os processos de planejamento, contratação ou mesmo marketing. A despeito do aparente uso ainda incipiente, a maioria dos respondentes afirmou que a tendência da modelagem deve se intensificar, e apontaram a melhoria da competitividade e da qualidade do trabalho como principal motivo para investir nas ferramentas de tecnologia de informação (VTT, 2007).

Em recente evento que reuniu participantes de vários setores da indústria da construção, Martin Fischer apontou como resultados comuns a vários projetos que utilizaram BIM o aumento da produtividade no canteiro de 20 a 30%, a redução das solicitações de situação e pedidos de modificação em dez vezes ou mais, o aumento expressivo do engajamento dos *stakeholders* e a consideração de muito mais opções de projetos a partir de mais perspectivas sem aumento de custos e tempo de projeto. Entretanto, a BIM tem sido utilizada, na maioria dos casos, em fases isoladas do processo de construção, e o seu maior desafio atualmente é a integração dessas fases. Cada setor da construção representado no evento colaborou com uma perspectiva diferente sobre a BIM e as demais barreiras a serem superadas. A conclusão geral foi que uma maior adoção da BIM esbarra em dois problemas: a mentalidade individualista dos envolvidos, que tradicionalmente não consideram o benefício da colaboração durante os projetos, e a falta de ferramentas que transfiram dados entre as fases do desenvolvimento (HARTMANN e FISCHER, 2008).

Howard e Björk afirmam que pesquisas quantitativas como estas tem demonstrado repetidamente o pouco conhecimento dos respondentes a respeito da modelagem e de padrões para os modelos. Para identificar as barreiras e perspectivas para a tecnologia, os autores conduziram uma pesquisa qualitativa, junto a especialistas do mercado e da academia. Ao final de 2006, 18 especialistas de sete diferentes nacionalidades haviam respondido os questionários, entre arquitetos, engenheiros, empreiteiros e especialistas de TI. Com relação à possibilidade de criação de modelos abrangendo todo o ciclo de vida do edifício, a maioria dos respondentes relatou a falta de padrões e definições para os formatos de dados. Padrões para os modelos BIM ainda são mal divulgados e incompletos. As ferramentas precisam se

adequar melhor aos processos da indústria e alguns destes, por outro lado, precisam ser revistos em face às novas possibilidades tecnológicas. A questão da educação dos profissionais, e quais conhecimentos serão necessários para atuar com a modelagem de edifícios deve tornar-se cada vez mais preponderante, segundo os especialistas. Apesar das observações, todos concordaram nos benefícios da tecnologia, sendo o cliente apontado como principal beneficiário na maioria das respostas (HOWARD e BJÖRK, 2008).

4
Acesso ao Modelo Integrado do Edifício

O objetivo de acessar os modelos de edifícios é criar aplicações para processar os seus dados, que foram previamente estruturados em uma aplicação BIM CAD, e gerar novas informações a partir deles. Coons, ainda em 1963, observou que grande parte da utilidade dos sistemas CAD residia na sua abrangência sobre todos os processos de produção, da concepção ao produto acabado. Porém, prever todas as situações possíveis em um projeto e programar respostas para elas, mesmo que fosse possível, resultaria em um sistema CAD complexo e rígido demais para ser útil. Como alternativa, Coons propôs que os sistemas CAD fossem compostos por uma aplicação responsável pelas rotinas genéricas necessárias para a representação básica, associada a outras que realizariam tarefas mais especializadas (COONS, 1963). Da mesma maneira, Eastman, descrevendo os primeiros sistemas de modelagem, afirma que a melhor abordagem para garantir a sua flexibilidade e adaptação a diferentes sistemas construtivos é permitir a complementação posterior da aplicação principal por sub-rotinas programadas na mesma linguagem (EASTMAN, 1976). Esta abordagem continua sendo defendida, como demonstram o trabalho de Magdy Ibrahim *et al*. (IBRAHIM *et al.*, 2004) e o de Umit Isikdag (ISIKDAG *et al.*, 2007).

Modelos de edifícios são construídos sobre modelos de dados, que definem as estruturas lógicas onde as informações são armazenadas. Modelos de dados podem ter formatos proprietários ou neutros. Formatos proprietários são desenvolvidos e mantidos por fabricantes de software, e o seu objetivo principal é garantir a eficiência no acesso aos dados pelo aplicativo a que se destinam. Exemplos de formatos proprietários são o DWG ou o 3DS, que são formatos internos do Autocad e do 3D Studio Max, respectivamente (AUTODESK, 2008); ou o PLN, o formato proprietário do ArchiCAD, do grupo Nemetschek/Graphisoft (GRAPHISOFT, 2008e). Há também formatos proprietários criados com o propósito específico de exportar informações

entre diferentes aplicações de um fabricante, ou entre as suas aplicações e outras criadas para trabalhar em conjunto com elas. Um caso típico é o DXF (*Drawing Exchange Format*), criado pela Autodesk para promover conexões entre diferentes aplicações com o Autocad. DXF é um modelo simples e de fácil implementação, prováveis motivos da sua ampla aceitação e aplicação, em praticamente todos os segmentos industriais (FOWLER, 1996).

Formatos neutros são resultado do desenvolvimento em uma PDT, como descrito na subseção 2.3, e seu objetivo principal é manter a eficiência da transmissão de informações entre diferentes aplicações, independente do fabricante. Idealmente, considerando a natureza multidisciplinar do desenvolvimento de produto e o consequente uso de aplicações de vários tipos, novas aplicações deveriam incorporar o formato neutro por princípio. Do ponto de vista do desenvolvimento de aplicativos, essa abordagem no mínimo exclui a árdua tarefa de definição de um formato de dados eficiente e flexível, e no máximo poderia eliminar os processos de tradução de dados entre diferentes formatos, que são sempre propensos a erros e perdas de significado da informação. Gielingh, porém, identifica vários desafios que ainda limitam a disseminação dos modelos neutros, entre eles a própria definição do que seria um modelo "neutro", já que mesmo diferentes *Application Protocols* da STEP não são interoperáveis (GIELINGH, 2008). Isso sugere que a adoção plena dos modelos neutros na indústria da construção, caso venha a ocorrer, será precedida por uma fase de transição na qual os modelos proprietários permanecerão exercendo forte influência na modelagem de edificações. Por esse motivo, neste trabalho foram estudados métodos de acesso aos dois tipos de formato: o formato proprietário PLN, do ArchiCAD (GRAPHISOFT, 2008e), e o formato neutro IFC, desenvolvido pela *International Alliance for Interoperability* (IAI, 2008c).

Existem várias formas de acessar modelos de dados, e identificar as principais para os modelos utilizados foi a primeira etapa da preparação para os experimentos neste trabalho. Um esquema organizando os métodos de acesso identificados foi elaborado e é apresentado na figura 4.01.

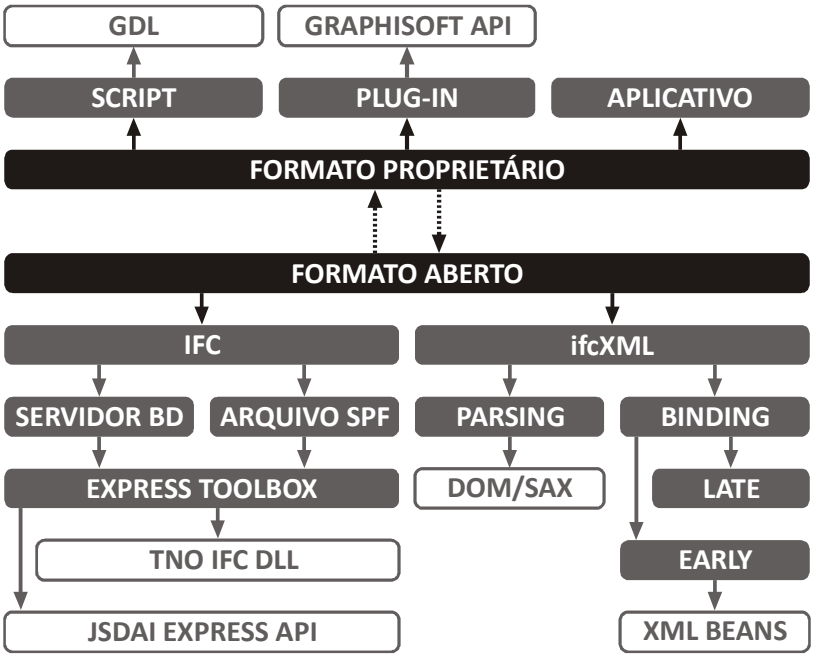


Fig. 4.01: esquema dos métodos de acesso ao modelo integrado do edifício utilizados neste trabalho.

A informação armazenada em modelos de formato neutro e de formato proprietários pode ser intercambiada por processos de mapeamento de dados. Neste trabalho foi utilizada a rotina de exportação do ArchiCAD, que possibilita gravar e abrir modelos de edifícios em formato IFC. Os três métodos para acesso ao modelo proprietário foram identificados como *script*, *plug-in* e aplicativo independente. ***Scripts*** são algoritmos criados para automatizar sequências de comandos de uma aplicação ou gerar novas informações a partir dos dados inseridos pelo usuário. Para avaliar esse método, duas aplicações foram desenvolvidas: objetos paramétricos representando paredes de blocos de concreto, e o EEQuant, uma rotina que quantifica energia e dióxido de carbono embutidos nos materiais de construção, a partir da associação entre os elementos que constituem o modelo do edifício e um banco de dados com índices de consumo. ***Plug-ins*** são pequenas aplicações criadas para complementar a funcionalidade ou a interface de uma aplicação BIM CAD. Eles funcionam exclusivamente em conjunto com a aplicação principal, da qual compartilham a interface e os procedimentos para manipulação de dados, através de bibliotecas de funções chamadas APIs (*Application Programming Interface*). A sua grande vantagem em relação aos *scripts* é um escopo de atuação maior, que oferece ao programador mais liberdade para interferir na aplicação principal. Para avaliar o método de acesso

ao modelo por *plug-in* foi criada uma aplicação para exportar elementos nativos do ArchiCAD para um sistema de análises físicas, chamado Mestre. Finalmente, **aplicativos independentes** possuem a sua própria interface e podem ser executados em separado da aplicação principal. Eles utilizam as APIs apenas para extrair as funções de acesso e manipulação dos dados de um modelo proprietário da aplicação em questão. Do ponto de vista da manipulação de dados e geração de informações, aplicativos independentes e *plug-ins* oferecem as mesmas possibilidades, já que se utilizam do mesmo método de acesso, a conexão via API.

Modelos de edifícios no formato de dados neutro estudado, IFC, podem se apresentar na forma de bancos de dados – gerenciados por um servidor de modelos – ou na forma de arquivos ASCII (*American Standard Code for Information Interchange*), um padrão de troca de informações por caracteres sem formatação. Neste trabalho não foram avaliados métodos de acesso via servidor de modelos, por dois motivos: primeiro, a complexidade envolvida na geração de um ambiente para testes demandaria um tempo que prejudicaria os demais experimentos. Mas, além disso, os dados obtidos a partir do modelo, seja ele apresentado como um banco de dados ou um arquivo ASCII são essencialmente os mesmos, e portanto as conclusões a respeito do acesso a modelos em arquivos podem ser generalizadas para o acesso de modelos em bancos de dados. Quando apresentados em arquivos, os modelos IFC podem ter dois formatos (embora apenas uma estrutura de meta-dados): podem ser descritos no formato SPF (STEP *Physical File*), ou então no formato ifcXML, que descreve a mesma informação dos arquivos SPF, porém através da estrutura de nós definida pela linguagem de meta-dados XML (W3C, 2008b). Para avaliar o acesso a modelos de edifícios em formato IFC SPF foram utilizadas bibliotecas API chamadas EXPRESS Toolbox. O acesso a modelos em formato ifcXML foi feito através de ferramentas de *parsing* (varredura dos nós XML) e *data binding* (mapeamento e associação das estruturas de dados para uma estrutura de classes).

Para identificar as vantagens e desvantagens dos métodos de acesso mostrados na figura 4.01, foram realizados diferentes experimentos. Para viabilizar a execução de experimentos com todos os métodos de acesso identificados, foram desenvolvidas aplicações muito simples, apenas para ilustrar mais facilmente as capacidades de cada

método. Estas aplicações não seguiram uma metodologia formal de desenvolvimento de softwares – não houve modelagem de processos, por exemplo. O desenvolvimento das aplicações e os resultados dos experimentos de acesso aos dados de modelos de edifícios são descritos nos próximos itens.

4.1 *Script*: objetos paramétricos representando paredes de blocos

Neste experimento foram estudadas as possibilidades oferecidas pelo acesso aos dados do modelo do edifício através de *scripts* para determinar o comportamento de objetos paramétricos que representassem paredes de blocos de concreto e automatizassem a geração de documentações técnicas.

4.1.1 Definição do problema

O potencial de racionalização da construção oferecido pelo uso da alvenaria de blocos de concreto depende de uma representação detalhada, de caráter executivo, que inclua a posição e o tipo de cada um dos blocos utilizados nas diferentes paredes da construção. Uma completa explanação sobre o método construtivo e suas especificidades pode ser encontrada em Wissenbach (1990), Pfeifer *et al*. (2001), Prudêncio Jr. *et al.* (2002), Ramalho e Corrêa (2003) e em Beall (2003). No Brasil, este requisito pode ser considerado um obstáculo para a disseminação mais efetiva desse sistema construtivo, já que a documentação projetual no país, tipicamente, traz poucos detalhes técnicos (FABRICIO *et al.*, 1999). Quando são utilizados desenhos no projeto de construções de alvenaria de blocos de concreto, a tarefa de produção da documentação detalhada demanda tempo e é propensa a erros. Muitas vezes o detalhamento do projeto sequer é realizado, ou então ocorrem modificações na versão já detalhada que demandariam muito trabalho para atualização dos desenhos, que ficam então desatualizados (SCHEER *et al.*, 2007).

A modelagem do edifício em um BIM CAD pode, em princípio, resolver parte desta situação, tanto pela melhor organização da informação quanto pela automação parcial da documentação do projeto. Entretanto, esses CADs não se adéquam bem a alguns aspectos técnicos e regionais do sistema construtivo em questão. O ArchiCAD, por exemplo, possui elementos nativos para representar paredes, mas apenas

genericamente, com os elementos constituintes das paredes sendo representados simbolicamente, com hachuras. Para ilustrar os inconvenientes dessa representação genérica para o sistema construtivo da alvenaria de blocos de concreto, considere-se a representação de duas paredes de blocos construídas com fiadas de prumo mostradas em planta e perspectiva na figura 4.02.

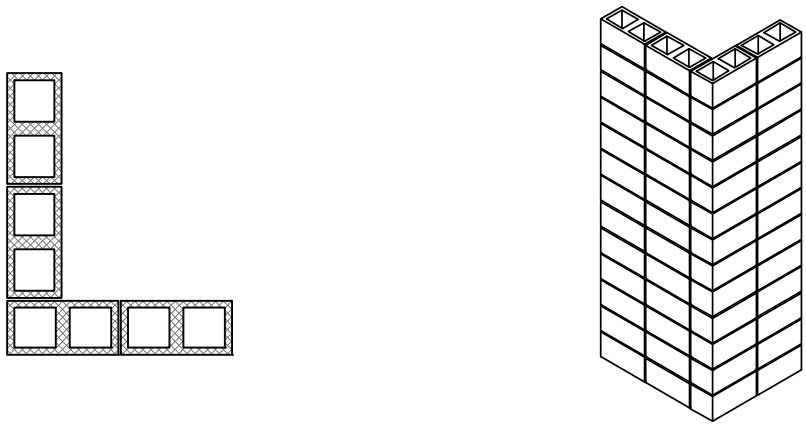


Fig. 4.02: representação em planta e perspectiva de duas paredes de blocos de concreto.

Os objetos nativos *wall* do ArchiCAD são objetos paramétricos que produzem representações automáticas em planta e perspectiva, com base no ajuste de parâmetros. Na figura 4.03a é mostrada a representação em planta obtida a partir da definição do material *concrete block* para as duas paredes. A definição do material possibilita calcular automaticamente a quantidade a ser utilizada nas duas paredes, a partir da avaliação dos seus parâmetros e do índice de consumo de blocos, por exemplo. A hachura em planta representa, simbolicamente, uma parede constituída de blocos de concreto. Por padrão, o ArchiCAD utiliza a convenção alemã DIN, mas também é possível criar hachuras que atendam as normas locais. A figura 4.03b mostra que a representação em planta pode ser aprimorada para incluir as duas camadas de reboco de 1 cm cada. O ArchiCAD identifica e corrige automaticamente o encontro entre os dois objetos *wall*, mantendo a coerência da representação das camadas que compõem as paredes.

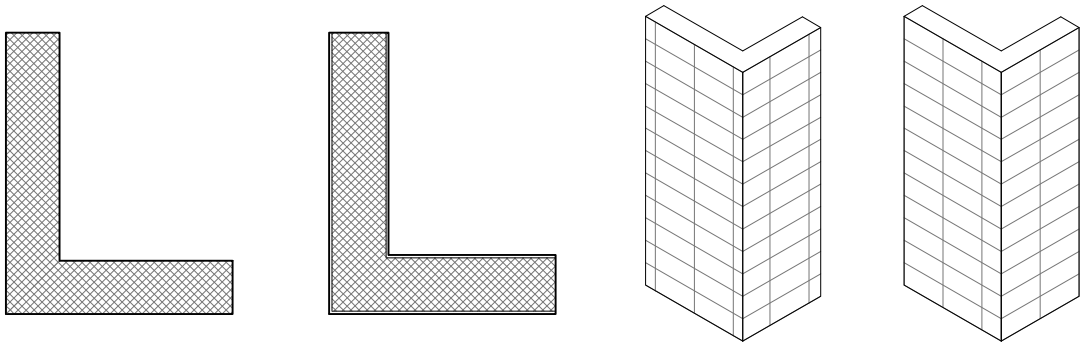


Fig. 4.03: Representações das paredes de blocos de concreto usando elementos nativos do ArchiCAD

Além de ajustar a representação em planta, também foi preciso selecionar uma hachura para simular as faces dos blocos de concreto nas vistas dos objetos *wall*. O resultado, mostrado na figura 4.03c, foi uma representação incorreta das faces dos blocos de concreto. Como a hachura é meramente simbólica, o sistema não julga necessário alinhar a origem do padrão de linhas que representa as faces dos blocos com o encontro das paredes. Uma operação adicional de alinhamento da hachura na perspectiva foi necessária, resultando na figura 4.03d.

Existem várias situações de projeto onde a representação simbólica obtida através das operações demonstradas acima seria suficiente. Por outro lado, considerando o paradigma da transmissão do modelo do edifício e do aprimoramento do seu conteúdo de informação durante as fases do projeto, em algum momento uma representação mais precisa e flexível seria exigida, antes que fosse iniciado o planejamento da construção. Para a alvenaria de blocos de concreto, a visualização individualizada dos blocos é especialmente útil nos projetos complementares e para o planejamento da construção, que utiliza plantas das fiadas e elevações das paredes de blocos. Os elementos nativos do ArchiCAD não fornecem a informação necessária para a geração dessa documentação. Scheer e outros autores, em estudo de caso conduzido em uma empresa de projetos especializada em construções com alvenaria de blocos de concreto, observaram que mesmo utilizando o ArchiCAD, os projetistas constroem muitas representações bidimensionalmente, do mesmo modo que fariam em um CAD baseado em primitivos geométricos, por falta de ferramentas adequadas (SCHEER *et al.*, 2007). Uma rápida leitura nas postagens da maior comunidade de usuários de ArchiCAD no Brasil revela outros exemplos de utilização de atalhos para solucionar

problemas para os quais os objetos nativos não fornecem soluções adequadas (ARCHICLUBE, 2008).

Um dos atalhos adotados pelos usuários nessas situações é o uso de símbolos bidimensionais que representam cada bloco, individualmente, em planta. Considerando apenas seis plantas de fiadas por pavimento – algo bastante corriqueiro – pode-se imaginar o vulto da tarefa de manter as diferentes representações atualizadas no caso de uma modificação no projeto. Outro atalho é a criação de um padrão de linha (*line template*) para desenhar símbolos dos blocos automaticamente acompanhando um percurso definido pelo operador, como mostrado na figura 4.04a. Porém o sistema não identifica situações onde blocos de ajuste ou meios-blocos são necessários, como mostra a figura 4.04b. E, pior ainda, em percursos curvos os blocos não são discretizados e rotacionados, e sim distorcidos gerando uma representação incorreta, como na figura 4.04c.

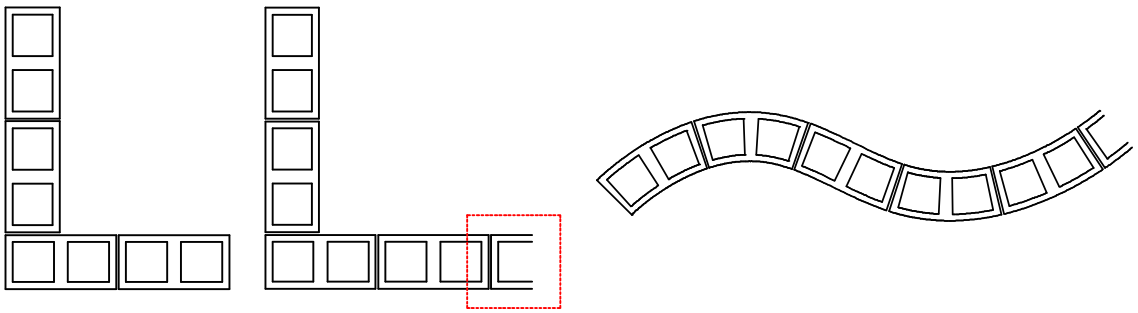


Fig. 4.04: representação das paredes de blocos com padrões de linhas (line templates).

A principal crítica a esses atalhos que baseiam-se em elementos bidimensionais é, porém, com relação ao fato de eles entrarem em conflito direto com princípios básicos da modelagem de produto: são tarefas que exigem reentrada manual de dados em várias vistas diferentes e não resultam em uma representação integrada. Para valer-se minimamente das possibilidades oferecidas pelo ArchiCAD, poderia ser adotada a abordagem de modelar blocos individualmente. Blocos modelados dessa maneira podem usar o objeto nativo *slab* para oferecer uma representação fiel das unidades, como mostrado na figura 4.05. Como são objetos tridimensionais, as representações para a documentação são geradas automaticamente. É possível

inclusive determinar a altura do plano de corte da planta, para gerar as diferentes plantas de fiadas.

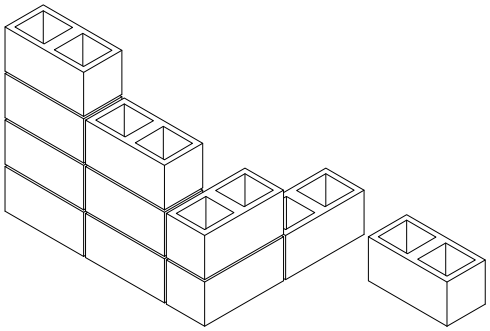


Fig. 4.05: representação das paredes de blocos com elementos slab configurados para assemelharem-se a blocos de concreto

Essa técnica, porém, está longe de ser recomendável. Modelar blocos individualmente pode ser útil para o estudo do encontro entre elementos, ou então demonstrar didaticamente as etapas da construção, ou ainda criar novas possibilidades baseadas no método construtivo, mas pode ser considerada uma prática inviável em projetos comerciais. Por mais que a representação resultante seja detalhada e correta, o tempo gasto pelos projetistas manipulando cada um dos milhares de blocos utilizados mesmo em construções de pequeno porte rapidamente superaria o benefício da utilização de um BIM CAD. Sacks e outros autores descrevem os efeitos negativos da modelagem de produto quando objetos muito pequenos ou demasiadamente detalhados são utilizados durante a definição do projeto, o que chamam de *bottom-up modeling* (SACKS *et al.*, 2004). Em resumo, os problemas da representação individual dos blocos de concreto são relacionados a:

- **Obter representações parciais**: os blocos de diferentes fiadas ou composições teriam que ser organizados em *layers* (camadas ou níveis) que precisariam ser ligados e desligados para facilitar a manipulação dos blocos.

- **Efetuar modificações no projeto**: uma vez que os blocos tenham sido dispostos, identificar e reposicioná-los pode se tornar uma tarefa ainda mais difícil do que utilizar elementos bidimensionais para representar as paredes.

- **Representar os componentes não discretizáveis**: os blocos modelados individualmente podem ter uma representação detalhada, mas as uniões entre os diferentes blocos não tem. Adicionar a representação da argamassa de rejuntamento ou revestimento cria um novo conjunto de desafios, para os quais a mesma discussão feita acima se aplica.

4.1.2 Abordagem proposta

A partir dos parâmetros dos objetos utilizados no ArchiCAD é possível criar *scripts* para automatizar a representação de paredes de blocos de concreto. Considerando a exposição da subseção anterior, o cenário paradoxal a ser enfrentado por essa ferramenta é operar em um nível mais alto do que a modelagem individual dos blocos de concreto, porém gerando automaticamente instâncias de nível mais baixo, utilizadas como representação detalhada. Eastman observa que durante o desenvolvimento de projetos, a informação torna-se articulada incrementalmente, e é inconveniente e pouco natural forçar os usuários a especificar todos os atributos de um elemento durante a sua definição inicial (EASTMAN, 1976). Os projetistas manipulam paredes abstraindo várias de suas características específicas. Elas são tratadas como painéis, e não como um conjunto de componentes que as constituem (independente de quais sejam). Assim, a partir de parâmetros dimensionais simples, a documentação detalhada deveria ser gerada automaticamente, de forma transparente para o usuário.

Para verificar essa possibilidade foi utilizada a linguagem interna de *scripts* do ArchiCAD, chamada *Geometrical Description Language* – GDL para construir um objeto paramétrico especializado. A GDL é uma linguagem estruturada, com a sintaxe simples e bastante semelhante à do Visual Basic. Há comandos GDL para a criação de todos os primitivos geométricos e também elementos construtivos nativos do programa. Uma revisão completa sobre a criação de objetos paramétricos no ArchiCAD pode ser encontrada no *Introduction to Object making* (NICHOLSON-COLE, 2004). A descrição dos comandos da GDL pode ser encontrada no manual técnico *GDL Reference Guide* (GRAPHISOFT, 2006; 2008b) e diretrizes para a criação de objetos paramétricos com a GDL podem ser encontradas no *Graphisoft GDL Technical Standards* (GRAPHISOFT, 2008f).

4.1.3 Desenvolvimento do experimento

Após a definição das características desejadas para as paredes de blocos, esboços guiaram o processo de programação. A GDL não requer ferramentas adicionais além do ArchiCAD – o código é criado em uma janela de texto, acessada a

partir do menu principal do programa. O código gerado é um algoritmo estruturado (fig. 4.06) que guia o sistema na criação de primitivos e sólidos geométricos de acordo com os variáveis que capturam os valores dos parâmetros especificadas pelo usuário. Esses parâmetros são dimensionais (por exemplo, espessura da parede, altura, tipo de bloco, etc) e de representação (espessura de linhas, hachura, cor, etc). Essas variáveis são primeiramente definidas no código, e então associadas a meta-parâmetros em um painel próprio (fig. 4.07).

```
if  (tipo = "BL" or tipo = "PA") and tipo3d = "Padrao" then
        block dimx/100, dimy/100, dimz/100
else
        if tipo = "BL" then
                if dimx = 39 or dimx = 29 then
                        prism_ 15, .19,
                                0, 0, 15,
                                x, 0, 15,
                                x, y, 15,
                                0, y, 15,
                                0, 0, -1,

                                .025, .025, 15,
                                x/2 -.025, .025, 15,
                                x/2 -.025, y - .025, 15,
                                .025, y - .025, 15,
                                .025, .025, -1,

                                x/2 + .025, .025, 15,
                                x - .025, .025, 15,
                                x - .025, y - .025, 15,
                                x/2 + .025, y - .025, 15,
                                x/2 + .025, .025, -1
                endif
```

Fig. 4.06: parte do código GDL que gera uma forma prismática representando um bloco de concreto.

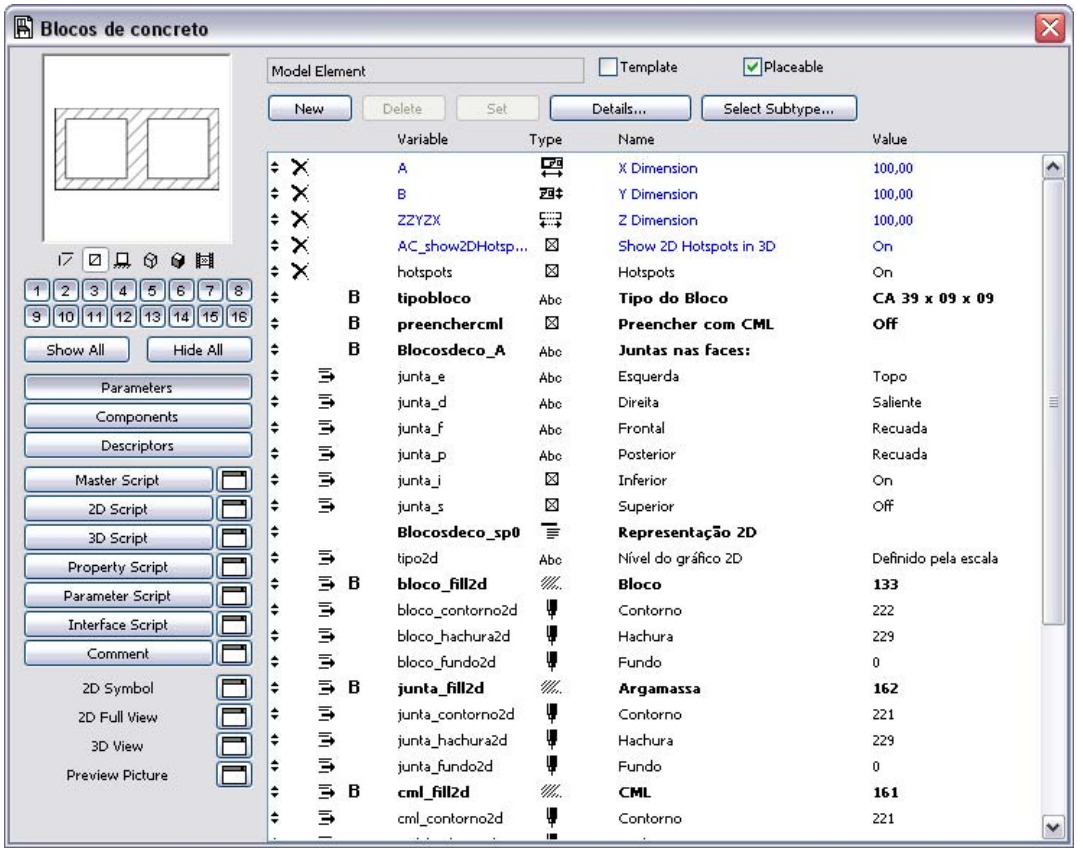


Fig. 4.07: janela de configuração de meta-parâmetros do ArchiCAD.

Resumidamente, o *script* cria uma matriz tridimensional de blocos para representar as paredes, a partir da avaliação dos parâmetros dimensionais. O código GDL completo encontra-se no Apêndice A. A utilização do objeto criado envolve duas interfaces: na janela da planta baixa do ArchiCAD é feita a inserção do objeto e a definição do comprimento (fig. 4.08). O número de parâmetros acessíveis por esta interface foi limitado para facilitar o uso do objeto nas fases iniciais do desenvolvimento do projeto. A segunda interface é a janela de configuração do objeto, onde parâmetros adicionais podem ser definidos (fig. 4.09). Esse painel de configurações é parte integrante da interface do ArchiCAD, e pode ser modificado a partir dos meta-parâmetros definidos anteriormente. A coordenação modular geométrica é essencial para o projeto de construções com alvenaria de blocos de concreto, e foi incluída no comportamento do objeto como uma restrição. Quando os objetos são dimensionados, o comprimento é aumentado em múltiplos da dimensão do bloco utilizado.

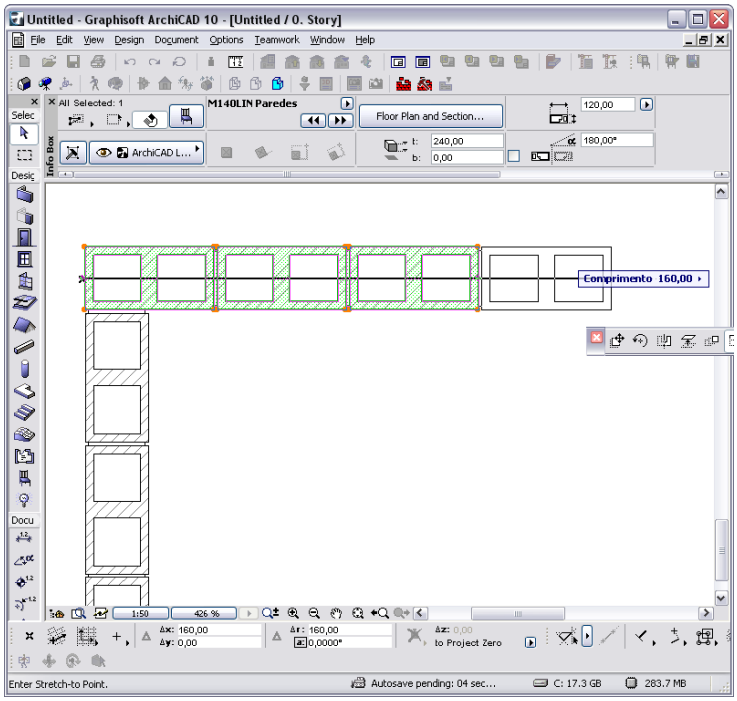


Fig. 4.08: Inserção dos objetos paramétricos em planta.

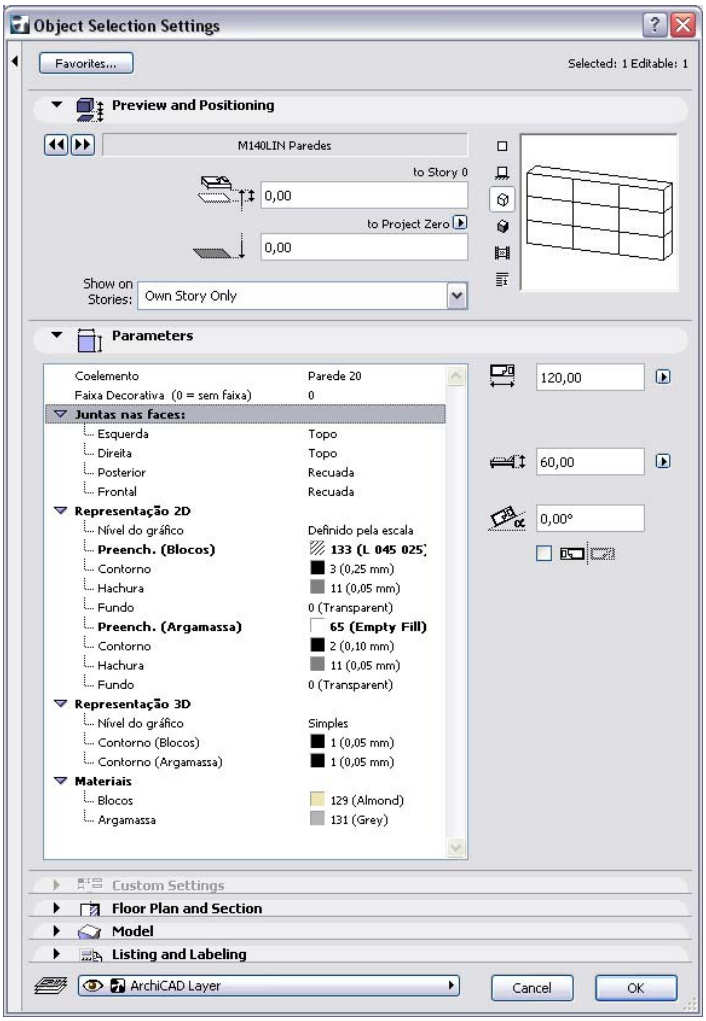


Fig. 4.09: painel de configuração de parâmetros adicionais.

Para permitir a representação adequada das paredes nas várias vistas geradas automaticamente (planta, corte, elevação, perspectiva), há parâmetros para controlar os elementos geométricos utilizados. Isso garante que o objeto comporte-se apropriadamente em cada situação. Por exemplo, pode-se configurar os objetos para que os blocos interceptados pelo plano de corte sejam hachurados, enquanto os outros são mostrados em vista (figs. 4.10a e 4.10b). Ao contrário da representação simbólica por hachuras, que gera um padrão simbólico de linhas, a matriz de blocos criada pelo *script* do objeto possibilita a identificação da real posição de cada bloco. Isso pode ser verificado nas figuras 4.10c e 4.10d, que mostram uma parede de blocos aparentes, onde uma fiada de detalhe arquitetônico é composta por blocos menores.

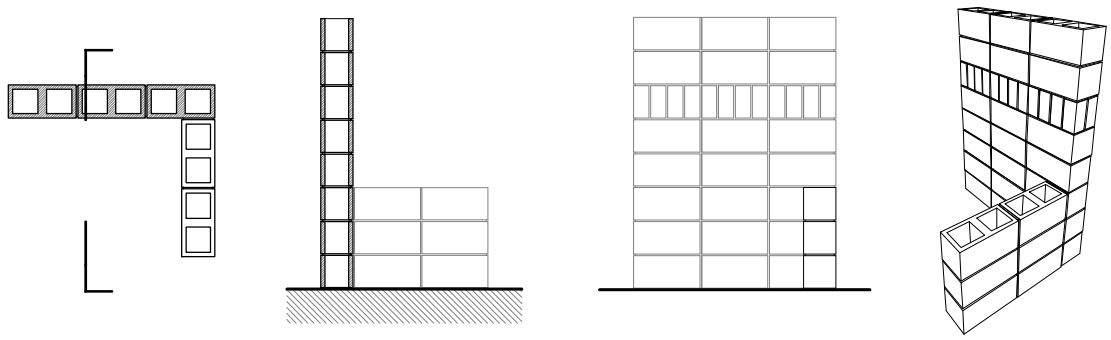


Fig. 4.10: representações das paredes de blocos de concreto geradas automaticamente pelo objeto paramétrico criado.

Outra funcionalidade programada foi o controle da quantidade de informação mostrada nas representações automáticas – ou seja, o nível de detalhamento desejado. Essa característica, chamada sensibilidade ao contexto, é uma das flexíveis possibilidades oferecidas pelo desenho por objetos paramétricos em vez de primitivos geométricos. Quando o contexto é alterado, todas as representações são automaticamente atualizadas, evitando a reentrada de dados. Na figura 4.11a são mostradas duas representações automáticas do mesmo objeto, em duas situações diferentes de escala de representação. A redução da quantidade de informação apresentada, em escalas menores, permite que o projetista concentre-se em aspectos mais genéricos do projeto. Na figura 4.11b é mostrada a representação em planta do mesmo objeto da figura 4.10a, porém em uma situação onde o usuário modificou a

altura do plano de corte da planta baixa, fazendo-o interceptar uma fiada mais alta. Essa função pode ser utilizada para gerar plantas das várias fiadas automaticamente.

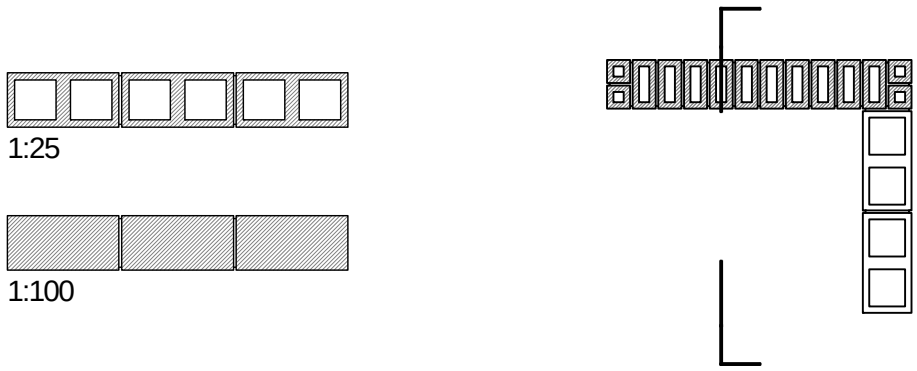


Fig. 4.11: representações automáticas geradas de acordo com o contexto do objeto

Scripts complementares foram criados para permitir que o objeto paramétrico representasse outros elementos construtivos baseados em blocos de concreto: peitoris, lintéis, parapeitos, paredes de blocos vazados, colunas, etc., dando origem a uma família de elementos organizados como um sistema construtivo. A associação das diferentes instâncias do objeto paramétrico especializado com objetos nativos do ArchiCAD permitem a representação completa do edifício. Na figura 4.12 é mostrado um exemplo dessa associação.

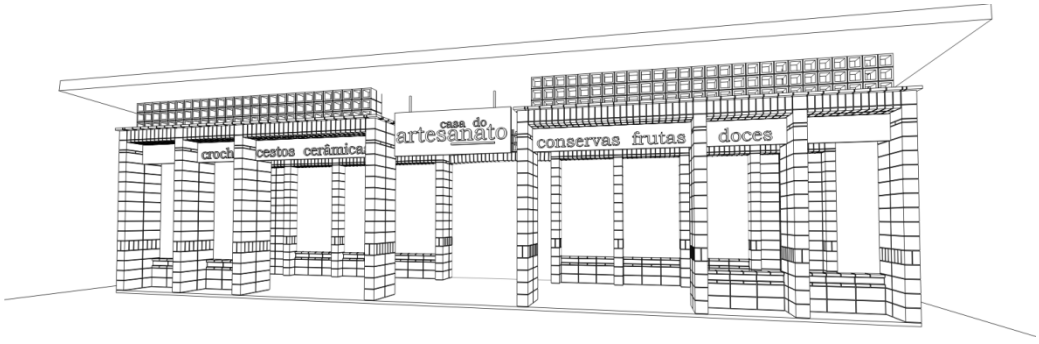


Fig. 4.12: modelo de edifício composto por várias instâncias do objeto paramétrico associadas a objetos nativos do ArchiCAD.

A representação gerada automaticamente a partir dos objetos paramétricos é precisa e detalhada. Na figura 4.13 é mostrada a planta de uma das fiadas do edifício da figura 4.12, extraída automaticamente a partir da configuração da altura do plano de corte da planta. A abordagem utilizada para criar os algoritmos responsáveis pelas

diferentes representações dos blocos poderia ser generalizada para incluir elementos textuais, como códigos de blocos e outras simbologias, completando o conjunto de informações necessárias para o planejamento da execução de obras de alvenaria de blocos de concreto.

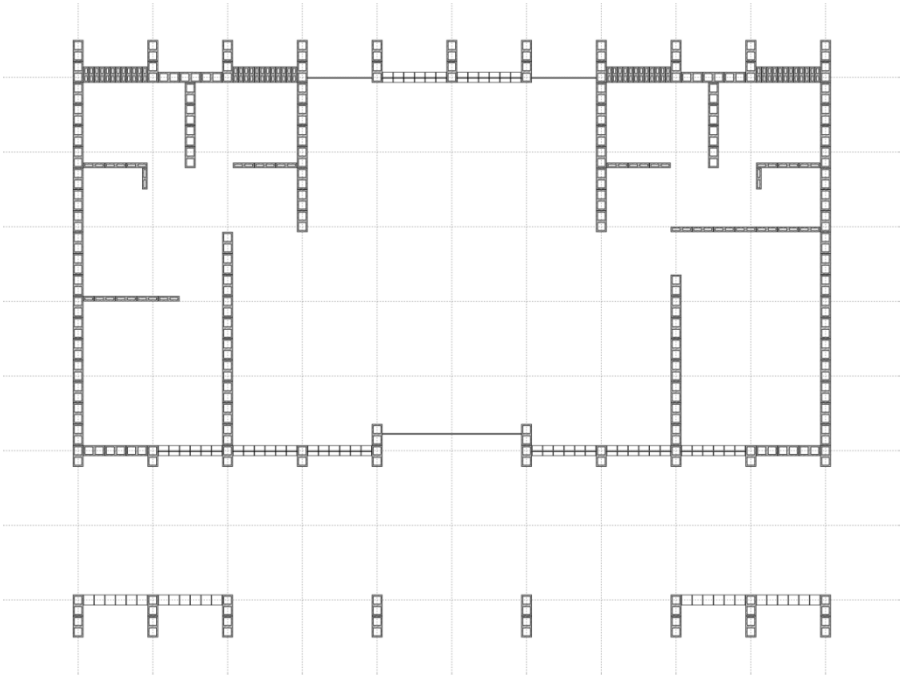


Fig. 4.13: planta de fiada extraída automaticamente a partir dos objetos paramétricos mostrados na figura 4.12.

As observações efetuadas durante o desenvolvimento e o uso do objeto paramétrico apresentado revelaram uma série de novos desafios impostos para a modelagem de edifícios de alvenaria de blocos de concreto (AYRES *et al.*, 2008a; AYRES *et al.*, 2008b). Várias características inerentes ao sistema construtivo, como a representação de peças estruturais, fiadas desencontradas, inserção de aberturas e sistemas complementares precisariam ser atendidas por uma versão aprimorada da ferramenta. O artigo no qual essas características foram relacionadas encontra-se no Apêndice B.

4.1.4 Discussão

Acessar o modelo proprietário do edifício via objetos paramétricos e *scripts* GDL é um procedimento relativamente simples, e há documentação disponível suficiente guiar o desenvolvedor iniciante. O desenvolvimento de objetos paramétricos

é facilitado pela utilização da interface e de procedimentos já existentes no ArchiCAD como, por exemplo, a geração automática de representações em diferentes vistas.

Uma desvantagem evidente, compartilhada por todos os métodos de acesso a modelos proprietários é a dependência da ferramenta criada em relação à aplicação principal, o ArchiCAD, e ao seu modelo de dados. Embora o modelo resultante possa ser exportado para o formato IFC, o desenvolvimento estaria condicionado às capacidades da rotina de mapeamento oferecida pelo fabricante.

Outra desvantagem é o escopo limitado oferecido pela GDL. Não foram identificadas nos manuais técnicos formas de relacionar duas instâncias diferentes de um objeto paramétrico, o que poderia ser muito útil para definir comportamentos específicos para estas situações. Por exemplo, várias das situações apresentadas no Apêndice B, como a resolução automática de encontros entre objetos representando paredes diferentes não poderia ser resolvida pela GDL. Como os objetos paramétricos criados a partir dela não são conscientes em relação aos outros objetos do modelo, o seu uso acaba limitado à representação de objetos isolados, cuja inteligência de contexto limita-se a identificar os parâmetros de representação selecionados pelo usuário e gerar respostas na forma de representações adequadas.

4.2 *Script*: EEQuant – quantificação de energia embutida e emissão de CO_2

Neste experimento foi avaliada a possibilidade de acessar os dados do modelo do edifício e associá-los a dados adicionais sobre o ciclo de vida dos materiais de construção mais utilizados no Brasil, reunidos no recente trabalho científico de Tavares (2006). A ferramenta desenvolvida, EEQuant, associa automaticamente esses dados, através de *scripts* e processos de quantificação disponibilizados pelo ArchiCAD. A informação resultante do uso da ferramenta permite avaliar instantaneamente, em fases iniciais do processo de projeto, parte do impacto ambiental causado pela utilização de diferentes materiais de construção.

4.2.1 Definição do problema

O aquecimento do planeta é um tema com repercussão cada vez maior na sociedade, e a indústria da construção é responsável por boa parte do consumo de energia e emissão de gases de efeito estufa (ROAF e DAY, 2001; ROAF, 2004; TAVARES, 2006). O impacto ambiental gerado pela indústria da construção nas economias em desenvolvimento é ainda maior do que nas desenvolvidas, pois grande parte da infra-estrutura ainda está sendo construída, e esse setor da indústria geralmente responde por uma proporção maior do PIB nacional (PLESSIS, 2001). A análise do impacto ambiental causado pelo edifício em todo o seu ciclo de vida torna-se cada vez mais importante. Por definição da ISO 14040, a análise do ciclo de vida considera as matérias primas e recursos energéticos consumidos em todos os processos envolvidos na produção, utilização e destinação final de um produto, bem como os seus impactos ambientais potenciais (TAVARES, 2006).

Os aplicativos que se propõem a analisar o ciclo de vida dos edifícios seguem um princípio comum: associam os dados específicos do edifício às bases de dados que contém informações sobre os processos de produção dos materiais utilizados na construção. Uma das bases de dados mais conhecidas é a Ecoinvent, um levantamento minucioso realizado através de uma iniciativa conjunta de institutos de pesquisa e órgãos governamentais suíços, que reúne dados sobre o ciclo de vida de mais de 4.000 produtos industriais (SCLCI, 2008). Outro exemplo é a base de dados utilizada pelo aplicativo LCADesign, que é equivalente à Ecoinvent, porém reunindo dados regionais da indústria australiana.

Um dos processos de entrada de dados utilizado por estes aplicativos de análise ambiental é o que se baseia em planilhas quantitativas. Essas planilhas são preenchidas pelo usuário com os dados extraídos manualmente do memorial descritivo do projeto do edifício a ser analisado. Dentre os principais aplicativos deste segmento estão o Eco-bat (FAVRE e CITHERLET, 2008) e o SimaPro (PRÉ, 2008). Uma desvantagem desse tipo de entrada de dado é a tarefa de transposição das informações entre diferentes aplicativos. Outro modo possível para a entrada de dados é o acesso ao modelo do edifício, e a extração automatizada de informações.

Essa abordagem aproveita as informações geradas em outras fases do processo de projeto, reduzindo drasticamente ou até evitando a reentrada de dados, que é sempre um momento propício ao surgimento de erros. Toda a etapa de leitura do memorial de materiais do projeto e a inserção das equivalentes quantidades no aplicativo específico poderia ser evitada, permitindo que o trabalho se concentre no processo de análise em si. O aplicativo LCADesign, desenvolvido pela Agência Nacional de Ciência da Austrália - CSIRO em parceria com indústrias do país, utiliza-se desta abordagem. Os modelos dos edifícios são importados de outros aplicativos através do formato IFC (TUCKER *et al.*, 2003; CSIRO, 2008).

4.2.2 Abordagem proposta

Além das duas alternativas para entrada de dados descritas na subseção anterior, é possível acessar o conjunto de informações necessárias para a análise de parte do ciclo de vida da edificação a partir da própria interface do BIM CAD, ainda durante as fases iniciais do desenvolvimento do projeto. Desse modo, o arquiteto poderia avaliar instantaneamente o resultado de suas escolhas em termos de impactos ambientais. Para isso, foi necessário acessar os dados do modelo a partir da interface do ArchiCAD e gerar diferentes algoritmos para quantificar automaticamente o volume de cada material utilizado e associar esses dados com informações sobre a produção de materiais de construção no Brasil, reunidos em um recente trabalho de Tavares (2006). Embora a base de dados utilizada seja muito menos volumosa do que os dois exemplos citados na subseção anterior, ela descreve as matrizes energéticas da produção dos materiais de construção mais utilizados no país, e o seu caráter regional permite avaliar com mais fidelidade o impacto ambiental da utilização dos diferentes materiais.

Além de verificar a possibilidade de acesso aos dados do modelo e sua associação com informações adicionais, a proposta da ferramenta EEQuant é conscientizar projetistas e estudantes de arquitetura a respeito dos efeitos ambientais das suas escolhas, fornecendo dados que fundamentem a decisão entre as diversas opções disponíveis para os sistemas que constituem o edifício, já nas fases iniciais de projeto. Para facilitar a sua aplicação, ficou definido que a ferramenta não deveria

gerar disrupturas no processo de projeto com o qual os projetistas estão habituados. Nesse sentido, o aspecto inovador da ferramenta está na abordagem adotada, que foi introduzir parte da capacidade das ferramentas de análise física de edificações diretamente no aplicativo CAD, com o qual os projetistas têm mais afinidade. Dessa maneira, foi possível agregar dados quantitativos do impacto ambiental da produção dos materiais de construção ao modelo da edificação, com mínimas modificações no processo projetual.

Na versão apresentada neste trabalho, a ferramenta EEQuant avalia de forma rápida, em fases iniciais de projeto arquitetônico, a quantidade de energia consumida e emissão de gás carbônico (CO_2) na fabricação e no transporte dos materiais de construção utilizados no edifício proposto.

4.2.3 Desenvolvimento do experimento

Para criar listas de quantificação de materiais, o ArchiCAD associa os parâmetros dimensionais dos objetos que compõem o modelo do edifício à bases de dados auxiliares que descrevem os componentes dos materiais utilizados. Essa associação torna o cálculo automático, pois o volume ou a extensão de cada elemento construtivo são obtidos pela combinação dos parâmetros dos objetos que os representam. Por exemplo, um objeto paramétrico representando uma laje, cujo parâmetro *material* foi definido pelo projetista como concreto: o processo de quantificação percorrerá a base de dados auxiliar e localizará os componentes que estão associados ao material, bem como os seus índices de consumo (areia, brita, cimento, água, aditivos, etc.) e calculará os volumes de cada um deles a partir do volume total do elemento construtivo, que é obtido a partir dos parâmetros dimensionais. O corpo principal da ferramenta EEQuant é justamente uma base de dados auxiliar, na qual foram inseridos (como componentes) os dados relativos ao consumo de energia e emissão de CO_2 na produção e no transporte dos materiais de construção. Essa inserção ocorreu em uma janela de edição do ArchiCAD (fig. 4.14). Os parâmetros dimensionais dos objetos paramétricos do ArchiCAD não incluem o peso e, por isso, os índices são calculados a partir do volume de material. Para fornecer o peso

total de cada material utilizado, foram inseridas na base de dados todas as densidades correspondentes.

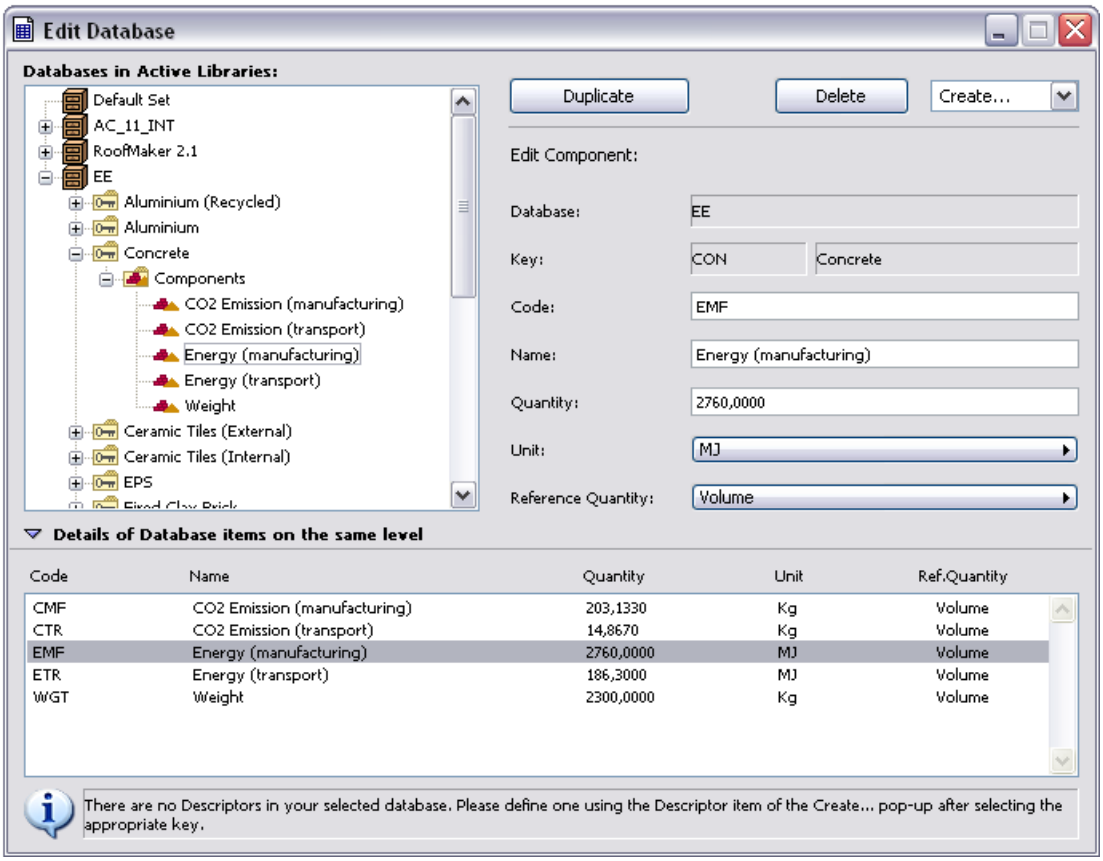


Fig. 4.14: janela de edição do ArchiCAD onde são criadas bases de dados auxiliares

A abordagem de quantificação por meio de associações entre bases de dados e os objetos paramétricos é um modo bastante dinâmico para se extrair dados do modelo do edifício. A partir de dados que são específicos de cada objeto (material, volume, posição, etc.), obtém-se as quantificações de quaisquer componentes que estejam presentes na base de dados ou que venham a ser inseridos posteriormente, sem que seja necessário modificar os objetos do modelo. Sem essas associações, seria preciso criar ou modificar parâmetros em cada um dos objetos que constituem o modelo, a cada vez que se necessitasse relacionar os elementos construtivos a informações adicionais como custo, horas de trabalho necessárias, prazo de entrega, ou – no caso da ferramenta EEQuant, emissões de CO_2 e energia consumida.

Além de flexibilizar e agilizar o processo de extração de dados, as associações entre objetos paramétricos e bases de dados auxiliares também facilitam o processo de modelagem em si. Por exemplo, uma parede de blocos de tijolo 9x19x19 com

reboco nas duas faces pode ter seus componentes (tijolos, areia, cimento e cal) facilmente calculados a partir de uma associação, através de fórmulas matemáticas simples que consideram os índices de consumo e o volume total do elemento. Não é preciso modelar cada tijolo e cada camada de argamassa individualmente, o que demandaria um tempo muito longo e dificultaria as alterações posteriores, como exemplificado no experimento anterior (seção 4.1). Disso resulta o tipo de modelagem descrito por Sacks e outros autores como *top-down modeling* (SACKS *et al.*, 2004), que produz modelos mais leves, demandando menor capacidade de processamento, e que são facilmente manipulados, podendo ou não passar por uma fase posterior de detalhamento, no qual os elementos construtivos são decompostos em unidades menores.

Com a base de dados completa, foram geradas as associações para relacionar os elementos construtivos nativos aos materiais e componentes criados. No ArchiCAD, estas associações são gerenciadas por elementos especiais chamados *property objects* (fig. 4.15). A função destes elementos é permitir a combinação de vários itens de uma ou mais bases de dados em um "pacote" de características que é então associado ao objeto paramétrico. Por exemplo, um *property object*, criado para quantificar paredes de alvenaria pode conter itens de uma base de dados de materiais (tijolos, areia, cimento, cal), de uma base de dados de recursos (custo, horas de trabalho, equipamentos envolvidos), e de quaisquer outras bases criadas para acrescentar informações, como a EEQuant, que fornece as quantidades de energia consumida e emissão de CO_2 na fabricação dos materiais utilizados na parede. Sem os *property objects*, a associação envolveria relacionar individualmente cada item desejado ao objeto paramétrico, uma operação demorada e sujeita a erros por parte do operador.

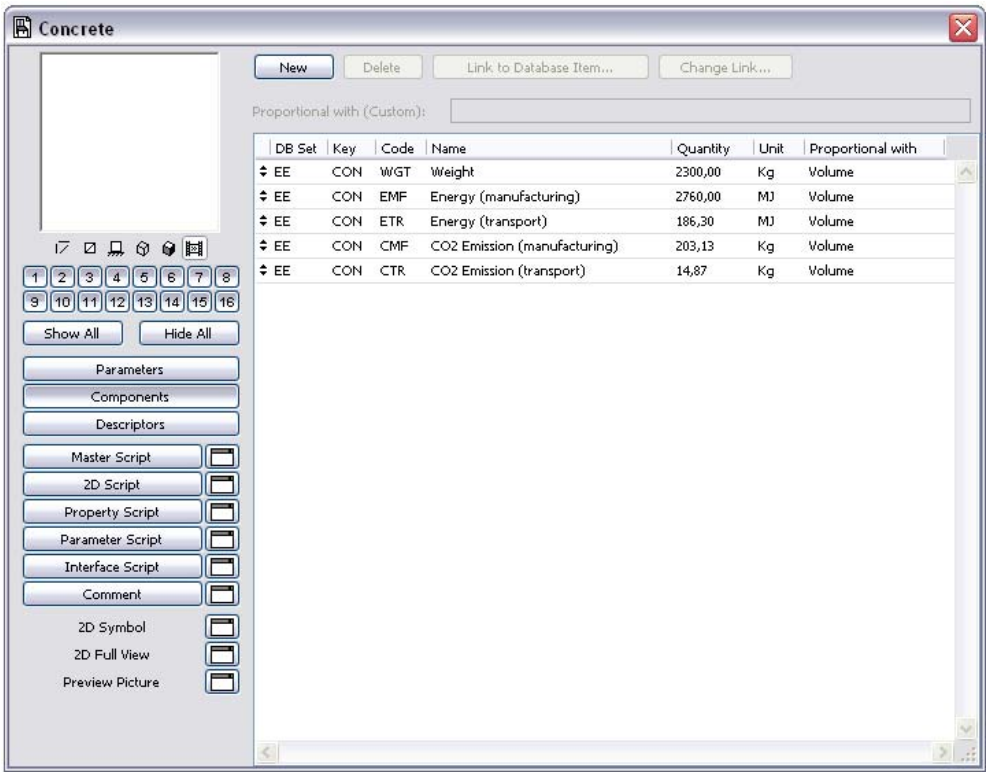


Fig. 4.15: janela de configuração do property object denominado "concrete" mostrando as relações entre os componentes e o volume do elemento construtivo.

Em um *property object* as associações podem ser diretas, requerendo uma simples seleção do item na base de dados auxiliar, ou indiretas, através de programação de *scripts* GDL. Algumas associações da ferramenta EEQuant foram resolvidas de forma direta, porque se referiam a materiais simples ou maciços, facilmente quantificados por volume (por exemplo, madeira, pedra, gesso, poliestireno). Para outros materiais, entretanto, a associação direta não foi suficiente e foi necessária a criação de *scripts* em GDL. Um exemplo é o caso dos revestimentos: a associação direta considerava apenas a superfície total do elemento construtivo, somando todas as suas faces, quando o correto seria discriminar os tipos diferentes de revestimento aplicados a cada uma das faces.

Em outras situações, além da necessidade de se tratar os dados de maneira mais complexa, os *scripts* em GDL possibilitaram a criação de algoritmos capazes de se adaptar a diversas situações, reduzindo o número de *property objects* utilizados e facilitando o uso da ferramenta EEQuant. Essa situação é exemplificada pelo caso das paredes compostas por várias camadas de materiais, como as de alvenaria: além de discriminar separadamente a quantidade de materiais em cada camada, era necessário

tornar a quantificação flexível para aceitar novas composições, por exemplo, com diferentes espessuras ou materiais de reboco e diferentes tipos de tijolos ou blocos. Parte de um desses *scripts* é mostrada na figura 4.16, bem como a janela de edição na qual são criados. Os *scripts* criados para a ferramenta EEQuant encontram-se no Apêndice C deste trabalho.

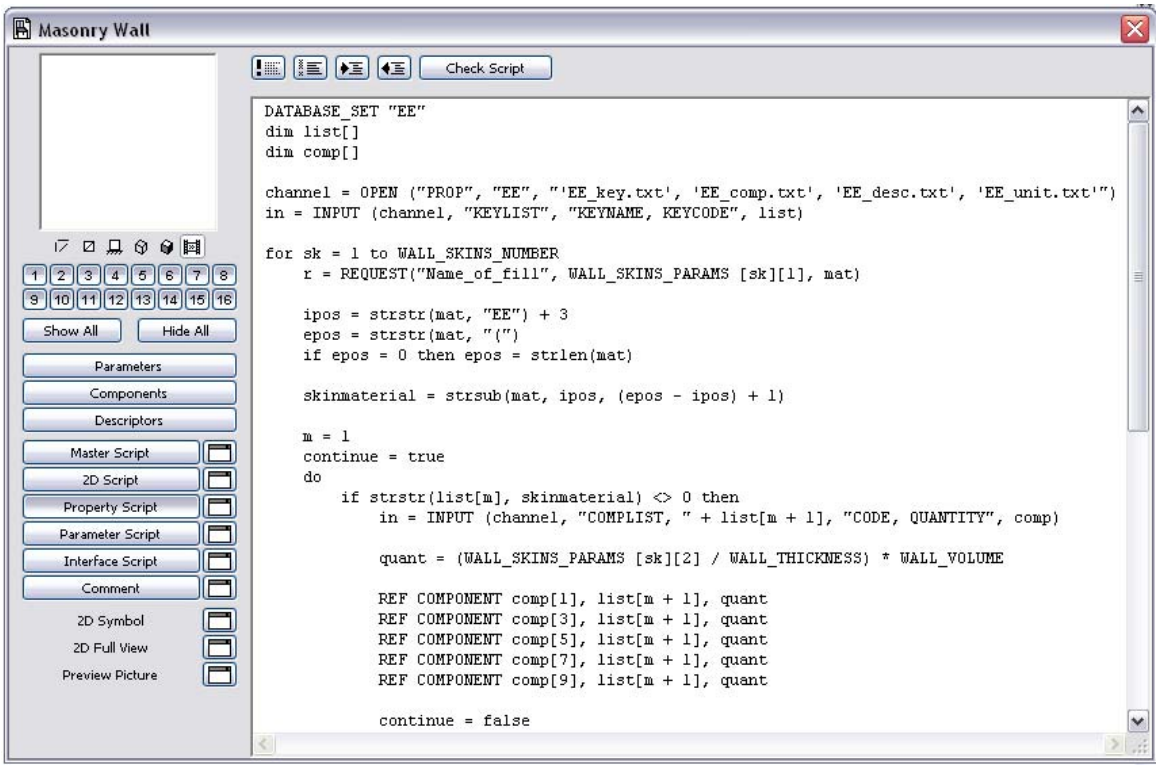


Fig. 4.16: parte do script GDL utilizado no property object "Masonry Wall", responsável pela discriminação dos materiais nas diferentes camadas de paredes de alvenaria.

A etapa seguinte no desenvolvimento da ferramenta foi a definição das associações entre os objetos paramétricos nativos e os *property objects*. O ArchiCAD fornece duas possibilidades de associação: automática e manual. Na forma automática são criadas regras de associação, que relacionam todos os objetos paramétricos que atenderem um determinado conjunto de critérios a um *property object* (e, conseqüentemente, aos itens correspondentes na base de dados auxiliar). Os critérios a serem atendidos podem ser, por exemplo, o tipo do elemento construtivo representado pelo objeto, o seu *layer*, cor do contorno, preenchimento, material, nome ou código de identificação. O ArchiCAD gerencia regras de associação a partir de uma janela de edição, onde são mostrados os critérios e os *property objects* correspondentes (fig. 4.17).

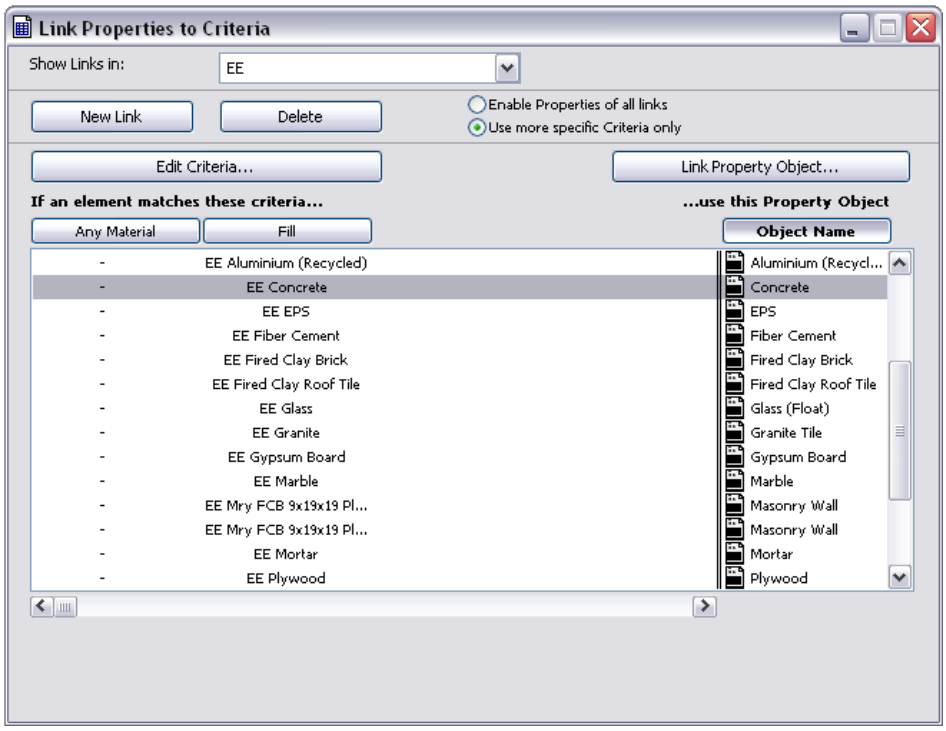


Fig. 4.17: janela de gerenciamento das regras de associação automática.

Para associar elementos construtivos aos componentes da base de dados auxiliar, a ferramenta EEQuant se utiliza de associações automáticas cujos critérios são a correspondência com o material de preenchimento ou revestimento do objeto paramétrico. Por exemplo, o *property object* "*Concrete*", que reúne os itens da base de dados auxiliar relativos à produção dos materiais que constituem o concreto, é associado automaticamente a todos os objetos paramétricos cujo parâmetro *fill* (preenchimento) for "*EE Concrete*", independente de quaisquer outros parâmetros do elemento construtivo (fig. 4.18a). Por sua vez, o *property object* "*Ceramic Tiles (External)*" é associado automaticamente a qualquer objeto paramétrico que possuir "*EE Ceramic Tiles (CTE)*" como valor para o parâmetro *material* (revestimento) (fig. 4.18b).

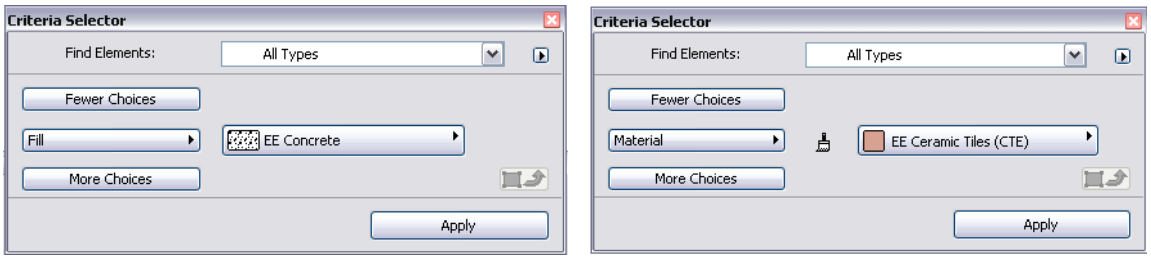


Fig. 4.18: configuração do critério para associação automática ao property object "Concrete" (esquerda) e ao property object "Ceramic Tiles (External)" (direita).

Existem situações, entretanto, para as quais o ArchiCAD não possibilita fazer associações automáticas. Para flexibilizar a quantificação de materiais compostos, a ferramenta EEQuant disponibiliza um *property object* que contém um *script* que quantifica cada material das diferentes camadas, sem que se precise associar cada nova composição a um *property object* diferente. Esse *script* se mostrou bastante útil, principalmente para fins didáticos, quando é preciso verificar qual é o efeito ambiental causado, por exemplo, ao se aumentar a espessura da camada de reboco nas paredes de alvenaria. Porém, as regras de associação automática do ArchiCAD não permitem que se crie um critério do tipo "todos os objetos que possuam material de preenchimento composto", sendo necessária a associação individual de cada material de preenchimento, o que inutilizaria o princípio de flexibilidade do *script*. Por isso, o *property object* "*Composites*", que calcula as quantidades de qualquer elemento construtivo composto por composição de materiais, deve ser associado manualmente, através da janela de configuração do elemento construtivo nativo (fig. 4.19).

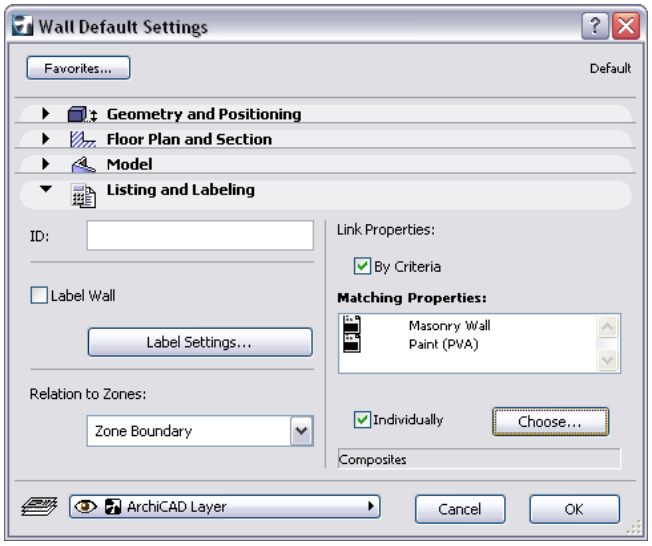


Fig. 4.19: associação manual do property object "Composites" na janela de configuração de um elemento construtivo nativo (nesse caso, wall).

A última etapa do desenvolvimento foi a configuração da saída de dados do processo de quantificação. Para isso foi utilizada a função *List Scheme* (esquema de listagem) do ArchiCAD, que permite configurar o formato da lista de quantificação, e também criar critérios para a inclusão seletiva de elementos construtivos nativos. Com esses critérios, pode-se quantificar apenas os elementos que representem lajes, ou então que sejam feitos de concreto, ou que estejam situados no andar térreo, etc. Também é possível configurar a lista para que mostre apenas as quantidades de itens que façam parte de uma determinada base de dados, nesse caso, a base de dados auxiliar da EEQuant. A janela de configuração do conteúdo da lista de quantificação é mostrada na figura 4.20.

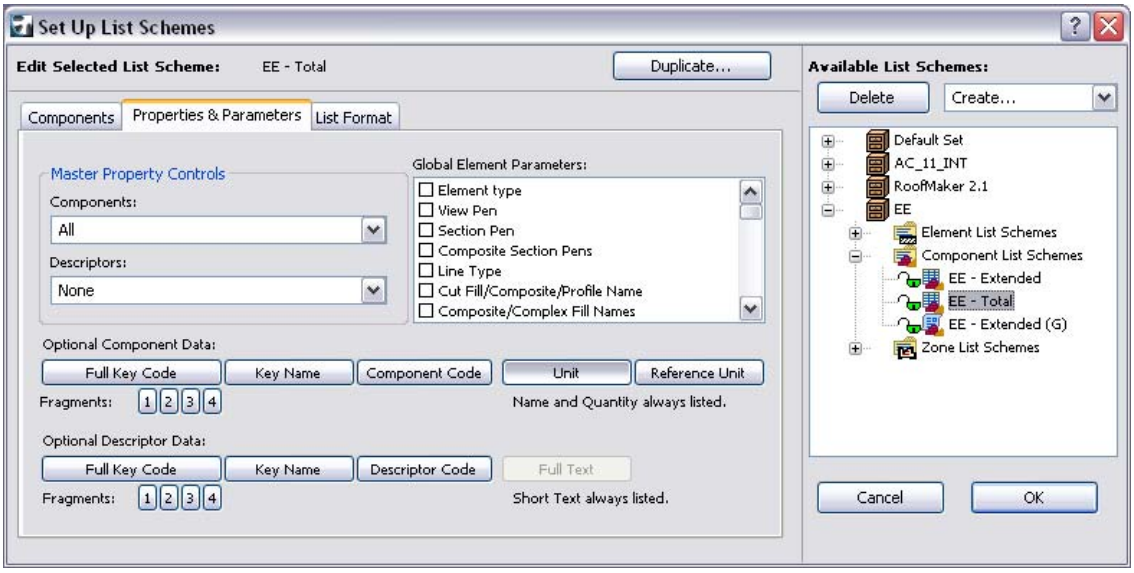


Fig. 4.20: configuração do conteúdo a ser apresentado em uma lista de quantificação na base de dados EEQuant.

4.2.4 Utilização da ferramenta

A utilização da ferramenta EEQuant apóia-se no processo de trabalho típico da modelagem do edifício no ArchiCAD, que por sua vez é bastante semelhante com os principais BIM CADs disponíveis. Durante a criação dos elementos construtivos, o projetista deve selecionar os materiais que compõem a base de dados da EEQuant, identificados pelas letras "*EE*" no início do nome (fig. 4.21). Como explicado na subseção anterior, elementos que não possuam esses materiais não serão associados à base de dados, e, conseqüentemente, não serão incluídos na quantificação.

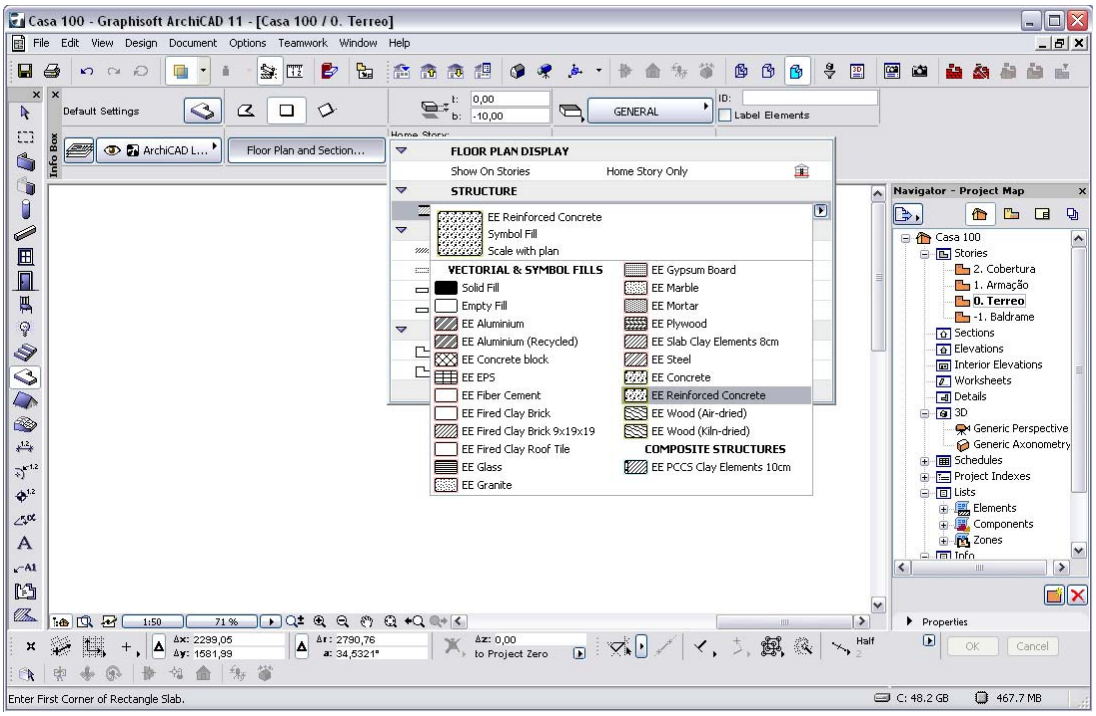


Fig. 4.21: seleção do material de preenchimento durante a criação de um objeto que irá representar uma laje.

Os materiais de preenchimento e revestimento selecionados podem, obviamente, ser modificados após o processo de criação, permitindo que o projetista verifique diferentes possibilidades e os impactos ambientais de cada uma delas. A modificação dos materiais é feita na janela de configuração de parâmetros dos objetos do ArchiCAD (fig. 4.22). A substituição dos materiais, desde que seja por outros da base de dados EEQuant, atualiza automaticamente as associações. Para o caso dos revestimentos, apenas as faces com materiais cadastrados na base de dados serão consideradas. Faces com demais materiais ou com o material "*EE Do Not Quantify*", criado especialmente para estas situações, não terão suas superfícies quantificadas.

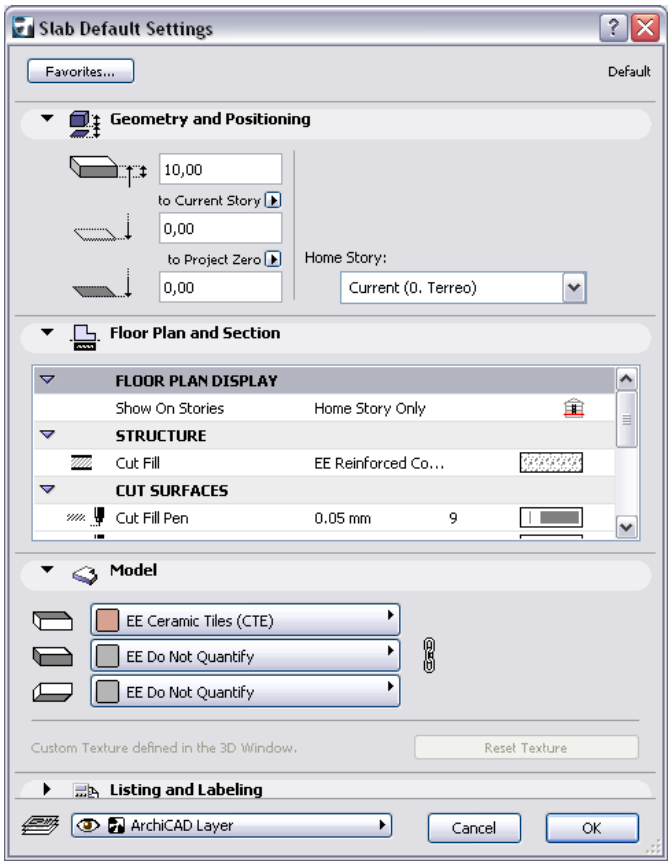
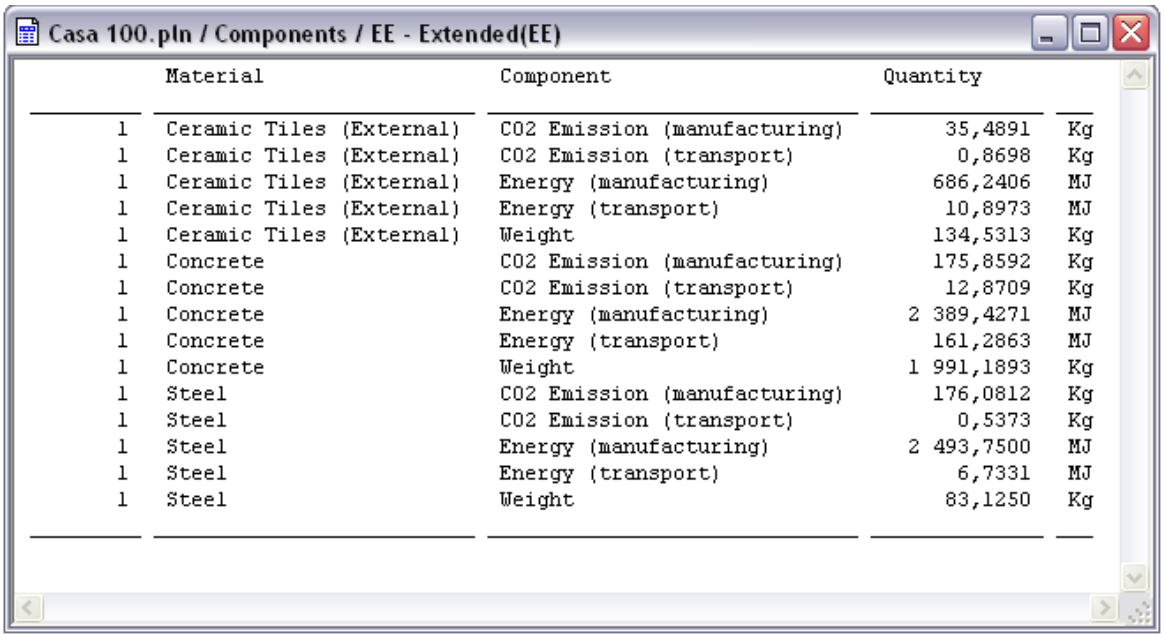


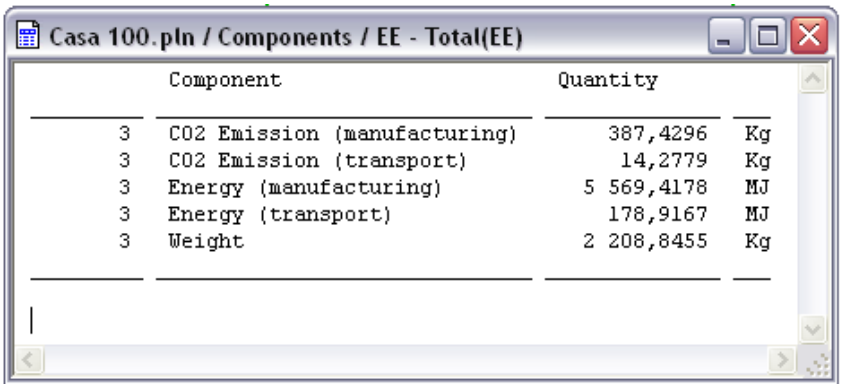
Fig. 4.22: janela de configuração dos parâmetros de um objeto representando uma laje.

A etapa seguinte consiste na definição do escopo de elementos a serem quantificados, selecionando os objetos desejados através das ferramentas do ArchiCAD. Caso nenhum objeto seja selecionado, serão quantificados todos os objetos que constituem o modelo do edifício. Selecionar um único elemento para fazer a quantificação também é uma maneira didática de se verificar parte dos efeitos ambientais causados pelo material escolhido. Por exemplo, analisando uma laje de 100 x 100 cm um aluno de arquitetura poderia verificar os níveis de energia consumida e CO_2 emitido na fabricação dos materiais demandados para cada metro quadrado do pavimento do edifício que utilizar o referido elemento construtivo. Foram desenvolvidos dois modelos básicos de listagem: a estendida, que mostra os índices de consumo de energia e emissão de CO2 para cada material (fig. 4.23) e a resumida, que mostra apenas os totais dos índices (fig. 4.24). Essas listas são acessíveis a partir da interface padrão do ArchiCAD, e podem ser copiadas e exportadas para planilhas eletrônicas.



	Material	Component	Quantity	
1	Ceramic Tiles (External)	CO2 Emission (manufacturing)	35,4891	Kg
1	Ceramic Tiles (External)	CO2 Emission (transport)	0,8698	Kg
1	Ceramic Tiles (External)	Energy (manufacturing)	686,2406	MJ
1	Ceramic Tiles (External)	Energy (transport)	10,8973	MJ
1	Ceramic Tiles (External)	Weight	134,5313	Kg
1	Concrete	CO2 Emission (manufacturing)	175,8592	Kg
1	Concrete	CO2 Emission (transport)	12,8709	Kg
1	Concrete	Energy (manufacturing)	2 389,4271	MJ
1	Concrete	Energy (transport)	161,2863	MJ
1	Concrete	Weight	1 991,1893	Kg
1	Steel	CO2 Emission (manufacturing)	176,0812	Kg
1	Steel	CO2 Emission (transport)	0,5373	Kg
1	Steel	Energy (manufacturing)	2 493,7500	MJ
1	Steel	Energy (transport)	6,7331	MJ
1	Steel	Weight	83,1250	Kg

Fig. 4.23: lista estendida, mostrando os índices de consumo de energia e emissão de CO2 na fabricação de cada um dos materiais utilizados.



	Component	Quantity	
3	CO2 Emission (manufacturing)	387,4296	Kg
3	CO2 Emission (transport)	14,2779	Kg
3	Energy (manufacturing)	5 569,4178	MJ
3	Energy (transport)	178,9167	MJ
3	Weight	2 208,8455	Kg

Fig. 4.24: lista resumida, mostrando apenas a totalização dos índices.

O processo para quantificar objetos paramétricos de um modelo existente – por exemplo, criado em outro BIM CAD e importado para o ArchiCAD via IFC é basicamente o mesmo. Pode-se proceder a análise de um modelo de edifício já existente de três formas básicas:

1. Substituindo os materiais de preenchimento e revestimento de cada elemento construtivo pelos materiais que compõem a base de dados EEQuant – o modo mais recomendado, por ser menos propenso a erros por parte do operador;

2. Associando os materiais de preenchimento e revestimento existentes aos property objects da EEQuant, criando associações automáticas – um modo ágil, porém que requer maior experiência por parte do operador;

3. Associando manualmente cada objeto paramétrico representando elementos construtivos do modelo – um modo rápido, que se presta a quantificações menores, de poucos elementos, mas desaconselhado em modelos completos, porque pode gerar discrepâncias caso o operador omita algum dos elementos construtivos.

Como exemplo da aplicação da ferramenta, foi criado o modelo de uma pequena residência, mostrado na figura 4.25. Desse modelo foram extraídas automaticamente as listas de quantificação de energia consumida e CO_2 emitido na fabricação e transporte dos materiais. Trata-se de uma casa térrea de 7 cômodos, com área aproximada de 105 m², em um terreno de 12 x 30 m. O sistema construtivo escolhido é o mais utilizado no Brasil: estrutura de concreto armado com vedações de blocos de cerâmica vermelha. As paredes são revestidas com argamassa simples e pintadas com tinta PVA ou revestidas de peças cerâmicas (áreas úmidas). Os pisos são lastros de concreto, com revestimento cerâmico ou de madeira. O forro é composto por laje pré-fabricada com elementos de cerâmica e capeamento de concreto. A estrutura do telhado é de caibros e ripas, com pontaletes, e a cobertura é de telhas cerâmicas tipo francesa.

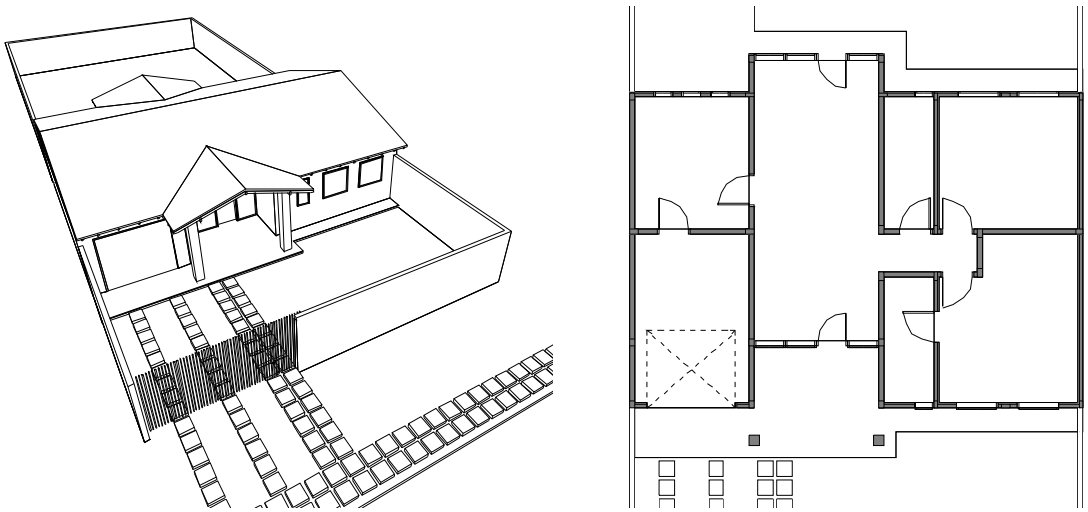


Fig. 4.25: modelo criado para exemplificar o uso da ferramenta EEQuant, em perspectiva e planta

A ferramenta demonstrou que mesmo uma casa relativamente simples como esta utiliza um volume de materiais de construção cuja produção consome consideráveis quantidades de energia: aproximadamente 328,5 GJ. O impacto ambiental fica mais evidente, entretanto, pela quantidade de CO_2 emitido: quase 24 toneladas. Como ilustração, de acordo com trabalho de Martins (2004), que considera a absorção média de carbono por árvores típicas da floresta atlântica brasileira, estima-se que seria necessário o plantio de 152 árvores, apenas para absorver o CO_2 emitido na fabricação e no transporte dos materiais utilizados. Não são consideradas nesse cálculo as demais fases do ciclo de vida da edificação (construção, utilização e demolição, por exemplo). As listas completas geradas pelo EEQuant para o modelo da residência do exemplo são mostradas no Apêndice D.

4.2.5 Discussão

O acesso aos dados do modelo digital e a sua associação com informações complementares mostrou-se viável e produziu interessantes resultados. A automatização conseguida pode ser generalizada para as mais diversas aplicações, sendo necessário apenas definir novos índices e incluí-los em bases de dados auxiliares. Por exemplo, poderia ser criada uma base de dados auxiliar que contivesse toda a informação disponível nas Tabelas de Composição de Preços para Orçamentos (TCPO), que já estão organizados em componentes e índices de consumo. Através dessa associação também poderia ser estendido o escopo de análise do ciclo de vida, já que as TCPO também incluem o consumo de energia de cada equipamento envolvido na produção dos elementos construtivos e na sua demolição (PINI, 2003). Até análises físicas preliminares poderiam ser realizadas através desse método de acesso aos dados do modelo do edifício. É possível, por exemplo, definir uma relação entre a massa do edifício, seus materiais e o volume dos cômodos, para determinar expeditamente a carga de condicionamento térmico necessária.

Como esse método de acesso está completamente integrado ao processo natural de projeto do ArchiCAD, foi observada grande agilidade e integração da ferramenta, que se tornou uma extensão das funcionalidades do CAD. A possibilidade de quantificação instantânea, sem que seja necessário um processo de transporte da

informação de um programa para outro, seja este processo manual ou automático, fornece ao projetista suporte para a tomada de decisões durante o projeto, permite que ele teste várias alternativas de projeto.

As desvantagens apontadas na subseção 4.1.4 também aplicam-se a este experimento, já que o acesso aos dados é praticamente o mesmo. A principal delas é que a ferramenta fica limitada à estrutura de dados e às interfaces disponibilizadas pela aplicação principal. O volume e a estrutura de dados necessários para avaliar o impacto dos processos de gestão de obra, por exemplo, extrapolam facilmente o escopo de um CAD arquitetônico, quer ele utilize modelagem ou não. Isto também se aplica ao caso das análises físicas, como por exemplo o desempenho térmico da edificação durante períodos pré-determinados de tempo. Esses tipos de análises mais elaboradas exigiriam o uso de um CAD 4D, que é capaz de produzir instâncias do modelo da edificação, incorporando a dimensão temporal.

4.3 *Plug-in*: aplicação para exportação de dados do ArchiCAD

A simulação do desempenho ambiental tem experimentado significativo desenvolvimento nas últimas duas décadas. Diferentes sistemas têm surgido, com diferentes graus de acessibilidade. Os mais acessíveis acabam servindo a um público mais amplo, e em especial como instrumentos didáticos na prática projetual. O sistema Mestre foi criado e vem sendo continuamente aprimorado pelo professor Aloísio Schmid, e atualmente executa análises térmicas, acústicas e lumínicas. Foi utilizado em quatro diferentes turmas do curso de Arquitetura da Universidade Federal do Paraná, como instrumento educativo, para demonstrar aos alunos a relação direta entre as decisões do projeto e as suas consequências no desempenho ambiental da edificação (SCHMID e AYRES, 2007). Uma descrição detalhada das funcionalidades do sistema Mestre pode ser encontrada nos trabalhos de Schmid (SCHMID, 2001; 2004; 2006).

Neste experimento foram verificadas as vantagens e desvantagens do acesso aos dados do modelo do edifício via *plug-in*. Uma aplicação foi desenvolvida para transformar informações do modelo do edifício criado no ArchiCAD para o formato nativo do sistema Mestre.

4.3.1 Definição do problema

O objetivo das atividades didáticas com as quatro turmas do curso de Arquitetura da UFPR, utilizando o sistema Mestre como instrumento didático, foi considerado atingido. Entretanto, o processo de modelagem das edificações pelos alunos foi considerada uma atividade desgastante. A entrada de dados no sistema Mestre é feita através de arquivos de texto ASCII, onde são inseridos parâmetros para orientar o programa na criação dos sólidos que representarão os elementos construtivos. Um exemplo de um arquivo de entrada de dados no sistema é mostrado na figura 4.26. Os diferentes elementos construtivos são criados a partir da operação de separação (*tokenization*) das várias *strings* que compõem o arquivo em comandos e atributos.

```
d    15   7    0 24 3600 -25 50 0 40 -3000 3000   17 26  0  0.25  0.5 0.5 1 1  0.4
CHAO
p    -5.0   5.0    0.0      0    90     10.0    0.25     10.0    3    2  1  25 chão
PAREDES
p     -5.0   5.0    0.0      0     0     10.0    0.15      3.0    4    2  0  25 norte
p      5.0   5.0    0.0     90     0     10.0    0.15      3.0    4    2  0  25 leste
p      5.0  -5.0    0.0    180     0     10.0    0.15      3.0    4    2  0  25 sul
p     -5.0  -5.0    0.0    270     0     10.0    0.15      3.0    4    2  0  25 oeste
TETO
p     -5.0   5.0    3.0      0    90     10.0    0.25     10.0    3    2  0  25 teto
fp  -10.0   10.0    0.0      0    90     20.0       20.0    6    2  0  25
TERRA
e      1.0      1.0  -6378000.0    6378000.0   7      terra
m    1.0  700   500 0.1 0.3 0.7 0.0 0.0 0.0  0.15 0.15 0.11 0.10 0.07 0.06 0.07       0   0   0    0.0  22  assoalho
m    1.0  700   300 0.1 0.3 0.4 0.0 0.0 0.0  0.04 0.04 0.04 0.15 0.29 0.52 0.59  0   0   0    0.0  2   carpet
m    1.0  700  2400 0.3 0.2 0.3 0.0 0.0 0.0  0.01 0.01 0.01 0.02 0.02 0.02 0.03  0   0   0    0.0  8   concreto
m    0.7  700  1200 0.1 0.3 0.3 0.0 0.0 0.0  0.01 0.01 0.01 0.02 0.02 0.02 0.02  0   0   0    0.00  4   alvenaria
m    1.0  700  2200 0.1 0.1 0.0 0.9 0.9 0.9  0.18 0.18 0.06 0.04 0.03 0.02 0.02  0   0   0    0.01 1   vidros
m    1.0  700   100 0.1 0.1 0.0 0.0 0.0 0.0  0.18 0.18 0.06 0.04 0.03 0.02 0.02  0   0   0    0.01 3   gramado
m    1.0  700  1000 0.3 0.2 0.1 0.0 0.0 0.0  0.20 0.20 0.10 0.10 0.06 0.05 0.00   0   0   0    0.0  6 planeta Terra
z 1000    0    0    0    0        23     23   23     23  0  0    0    0    0       0 0    0 21 1    ar externo
z 1000    0    0    0    0    100000 100000 1000000 100000  0    0    0    0    0       0 0    0 21 1    subsolo
z 1000    0    0 5000    0        8      8       8       8  0   0    0    0    0       0 0    0 21 1    inferior
X ================================================================================================================
ta 2 18.6 18.4 18.2 18.1 17.9 17.8 18.4 20.1 21.9 23.8 25.4 26.5 27.6 27.7 27.7 27.6 26.2 24.8 22.3 20.8 20.1 19.5 19.1 18.8
ta 6 2.3  1.8  1.3  0.8  0.5   0.2 -0.3 -0.6  0.8  3.5  6.6  9.1 11.1 12.0 13.0 13.3 13.1 11.8 9.4  7.3  6.3  5.1  4.4   4.0
ta 7 2.3  1.8  1.3  0.8  0.5   0.2 -0.3 -0.6  0.8  3.5  6.6  9.1 11.1 12.0 13.0 13.3 13.1 11.8 9.4  7.3  6.3  5.1  4.4   4.0
tma 9 -10 20
```

Fig. 4.26: arquivo de entrada de dados do sistema Mestre, a partir dos quais são construídas as representações dos elementos construtivos.

Como observado por Hoffmann e Joan-Arinyo (2002), criar modelos a partir de linhas de comando é uma tarefa pouco intuitiva para os projetistas, e deve ser evitada sempre que possível. De fato, a maioria dos alunos demonstrou dificuldades na criação do modelo, já que a maioria nunca teve qualquer contato linguagens de programação ou com sistemas computacionais baseados em comandos textuais. Além disso, completar o arquivo de texto que daria origem ao modelo era apenas o primeiro passo. Como a intenção da análise do desempenho ambiental é apontar deficiências e embasar modificações no projeto, havia sempre a necessidade de modificar o arquivo inicial, substituindo materiais, criando ou eliminando elementos construtivos. Se por

um lado a substituição de materiais era elementar, envolvendo a troca de um único atributo no arquivo-texto, a alteração da geometria (como por exemplo a criação ou eliminação de aberturas) mostrou-se morosa e foi evitada pelos alunos na maior parte das situações. Em avaliação não sistemática, os alunos reconheceram o valor didático da atividade, porém reivindicaram métodos mais fáceis para a entrada e a modificação dos dados que originarão o modelo.

4.3.2 Abordagem proposta

O sistema Mestre representa elementos construtivos parametricamente, de maneira muito semelhante ao ArchiCAD. Os vários parâmetros de cada elemento é que possibilitam a execução das análises (por exemplo, densidade, massa, índice de reflexão da luz, etc). Essa afinidade de modelos de dados sugeriu que o processo de tradução entre os formatos das diferentes aplicações seria viável. Além disso, a solicitação dos alunos por uma interface mais intuitiva poderia ser respondida transferindo-se o processo de construção do modelo para o ArchiCAD, e posteriormente exportando essas informações para o formato nativo do sistema Mestre, que se encarregaria de executar as análises.

O mapeamento completo das estruturas de dados que definem os diferentes objetos nos dois sistemas seria uma tarefa muito extensa, e interferiria no tempo necessário para a condução de outros experimentos deste trabalho. Por isso, foi decidido criar uma aplicação-piloto simplificada, inicialmente exportando apenas as paredes criadas no ArchiCAD para o sistema Mestre. Como o modelo de dados do ArchiCAD determina padrões que são muito semelhantes para todos os objetos, considerou-se que a experiência com a exportação das paredes possa ser generalizada para todos os outros elementos construtivos, de forma que a aplicação final seguiria a mesma lógica proposta.

4.3.3 Desenvolvimento do experimento

Para que aplicações especializadas possam ser criadas e utilizadas em conjunto com o ArchiCAD, complementando as suas funcionalidades, a Graphisoft oferece um conjunto de bibliotecas de vínculos para programação em C++, chamada *Graphisoft Application Programming Interface Development Kit*, ou simplesmente Graphisoft API (GRAPHISOFT, 2008d). A API é disponibilizada gratuitamente para uso não-comercial, e um cadastro junto à Graphisoft foi necessário para que um código que habilita o funcionamento de aplicações criadas com a API fosse disponibilizado.

Ao contrário da criação de *scripts*, o desenvolvimento de aplicações com a Graphisoft API não utiliza a linguagem GDL, e sim C++, que é a mesma linguagem na qual o ArchiCAD foi desenvolvido. Isso não garante só uma melhor adaptação da nova aplicação ao sistema principal, mas também viabiliza algoritmos muito mais complexos, de escopo mais amplo e processamento mais rápido do que seria conseguido com a melhor programação GDL. Também pode-se criar aplicações em Java utilizando a API, desde que sejam definidos métodos nativos para gerenciar a conexão com as funções das bibliotecas DLL. Essa abordagem não é encorajada pela Graphisoft, por ser mais propensa a erros de alocação de memória durante a execução. Os métodos disponibilizados pela API podem ser utilizados tanto para a criação de *plug-ins* como aplicações independentes. Pode-se, por exemplo, desenvolver um sistema de análises físicas completo em C++, utilizando funções de diversas bibliotecas – entre elas as da Graphisoft API. O resultado seria um programa completamente independente do ArchiCAD, que, no entanto, seria capaz de ler e gravar modelos no formato de dados proprietário da Graphisoft. As funções disponibilizadas pela Graphisoft API são razoavelmente simples para profissionais com conhecimento da linguagem C++. Uma referência completa destas funções pode ser encontrada na documentação fornecida pela fabricante (GRAPHISOFT, 2008c).

A abordagem adotada para o algoritmo criado foi percorrer todos os objetos *wall* do edifício em um *loop* principal, identificando os parâmetros dos objetos e gerando uma pilha de *strings* no formato nativo do sistema Mestre. O formato de dados dos modelos de edifícios no ArchiCAD é organizado em uma série de bases de

dados, uma para cada tipo de elemento construtivo nativo (*wall*, *slab*, *column*, *beam*, *window*, etc). Para obter a coleção de todos os objetos *wall* existentes no modelo, a base de dados correspondente é transferida para uma matriz temporária, que então pode ser percorrida elemento por elemento. A estrutura de dados do modelo do ArchiCAD é orientada por objetos, então para realizar a conversão de dados, basta chamar todos os parâmetros desejados e transpô-los para a nova posição – neste caso, dentro das *strings* que formarão os comandos do sistema Mestre. Além dos parâmetros geométricos, são processados também os parâmetros complementares *material* e *fill*, que correspondem ao material de revestimento e ao material do núcleo de cada uma das paredes, respectivamente.

O sistema Mestre não possui um comando para a criação de aberturas nas paredes. Janelas e portas são representadas como paredes mais estreitas, e com o material próprio (madeira e vidro, por exemplo). Do mesmo modo, vergas e parapeitos são representados como segmentos de paredes. Para o ArchiCAD, aberturas nas paredes são representadas pelos objetos *window* e *door*. Estes objetos possuem as suas próprias bases de dados, mas mantém um vínculo indissolúvel com as paredes nas quais estão posicionados. Durante o laço (*loop*) principal do programa, uma rotina especial é chamada para discretizar e segmentar as paredes sempre que uma abertura associada a ele é localizada. O resultado é um conjunto de paredes cortadas pela extensão das laterais verticais das aberturas, como demonstrado na figura 4.27.

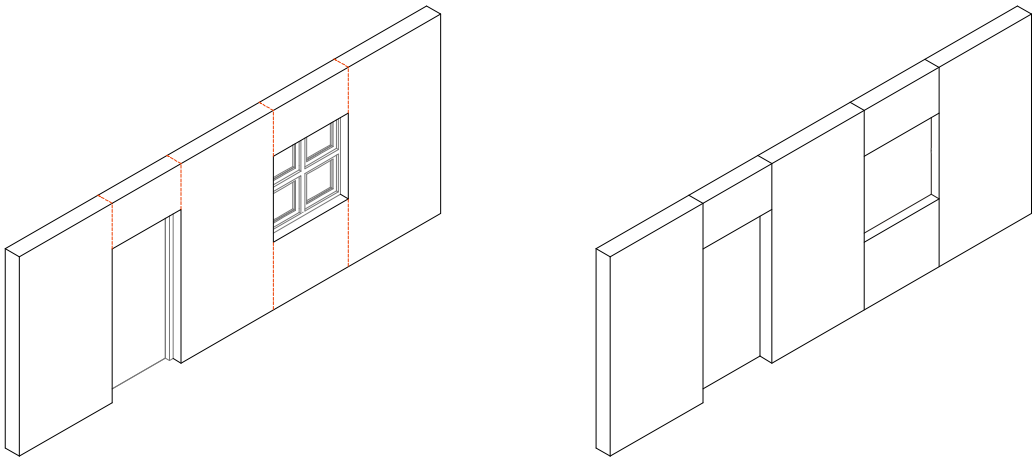


Fig. 4.27: diagrama demonstrando a operação de segmentação dos objetos wall.

Foi necessário também criar uma pequena função de ajuste da precisão dos números reais, que era diferente para os dois sistemas. Outra discrepância que precisou de tratamento foi o ângulo de rotação das paredes no eixo vertical: os dois sistemas adotavam origens diferentes para a medida.

Novos *loops* poderiam ser criados para realizar a exportação dos outros elementos construtivos nativos, e parâmetros adicionais podem ser extraídos por novas rotinas, utilizando a mesma abordagem. Quando a leitura dos objetos *wall* se completa, são acrescentadas *strings* para melhorar a visualização do arquivo texto e facilitar eventuais edições diretas.

A última etapa do desenvolvimento foi definir uma complementação para a interface *save as* (salvar como) do ArchiCAD. Desse modo, a partir da janela de perspectiva, o usuário acessa o menu e, entre as opções de formato de arquivo, surge o formato ".obj (Mestre)" para exportação. A aplicação de exportação é executada pelo ArchiCAD assim que o usuário clica no botão *save* da janela de diálogo *save as*. O algoritmo completo da aplicação encontra-se no Apêndice E deste trabalho.

A instalação da ferramenta é feita como qualquer outro *plug-in* do ArchiCAD. Uma pasta contendo a aplicação, com a extensão APX é copiada para a subpasta *plug-ins* contida na pasta de instalação do programa. Na figura 4.28 um conjunto de paredes criadas no ArchiCAD para demonstrar o uso da ferramenta é mostrado em planta e perspectiva. Na figura 4.29, o arquivo resultante do processo de exportação é mostrado.

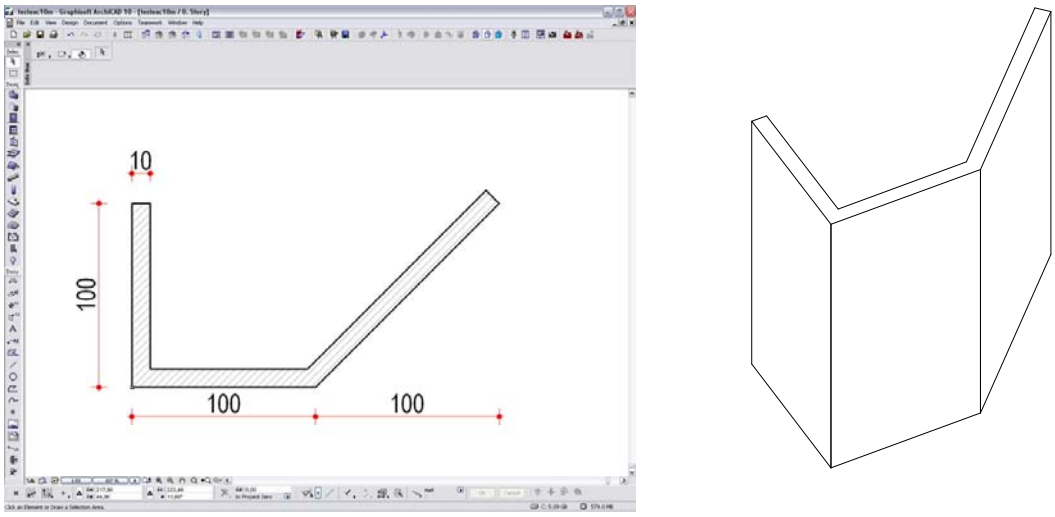


Fig. 4.28: conjunto de paredes criado no ArchiCAD para testar a exportação para o sistema Mestre.

```
//Arquivo exportado do AC10 para o programa Mestre
//
//Paredes
//
//x       y    z    azi  alt  larg      esp  h    mat  zf   zi   n
//
p 0.1     0    0    90   0    1         0.1  2    66   0    0    25   Parede 90
p 0       0    0    0    0    1         0.1  2    66   0    0    25   Parede 0
p 1       0    0    45   0    1.41421   0.1  2    66   0    0    25   Parede 45
//Fim do documento exportado
```

Fig. 4.29: arquivo de exportação gerado a partir do conjunto de paredes da figura 4.28.

4.3.4 Discussão

O acesso ao modelo do edifício via *plug-in* mostrou uma flexibilidade muito superior ao acesso via *script*. Um dos problemas apontados na subseção 4.1.4 – o escopo reduzido e a consequente impossibilidade de programar uma inteligência contextual mais refinada nos objetos – é facilmente resolvido pela possibilidade de acessar a coleção completa de objetos que constituem o modelo, e então fazer diferentes relacionamentos entre eles.

A programação em C++, ao contrário da feita em GDL possibilita algoritmos complexos, que podem conter funções especialmente desenvolvidas para cálculos matemáticos, análises físicas e geração de interfaces, disponíveis na Internet. Através da criação de *plug-ins* também pode-se acessar e modificar partes da interface do ArchiCAD. É possível criar novos menus, novos conjuntos de botões e ferramentas e

até novas janelas de diálogo com o usuário. Embora a criação de *plug-ins* seja muito mais complexa do que a criação de *scripts* GDL, a documentação fornecida pela Graphisoft facilitou bastante o desenvolvimento da aplicação neste experimento. Uma outra possibilidade interessante é a utilização da Graphisoft API para permitir que uma aplicação independente consiga ler e gravar modelos no formato proprietário da fabricante. Outras APIs de outros fabricantes poderiam ser utilizadas em conjunto para permitir a leitura e gravação de outros formatos de modelos de edifícios, flexibilizando o sistema criado.

Assim como na criação de *scripts* GDL, a grande desvantagem desse modo de acesso ao modelo é o fato da aplicação resultante não ser portável. Ela é dependente da aplicação principal, o que limita a escolha por parte dos usuários ou obriga o programador a criar várias versões de um mesmo *plug-in*, para os principais BIM CADs existentes, já que as ferramentas são incompatíveis entre si.

Outra desvantagem é a necessidade de atualizações constantes da ferramenta, sempre que uma nova versão do ArchiCAD for lançada. Recentemente a fabricante informou, através do fórum de desenvolvedores de *plug-ins*, que um novo e aprimorado conjunto de conexões entre o núcleo da aplicação principal e os *plug-ins* foi desenvolvido para a versão 10 do sistema, tornando as próximas atualizações de *plug-ins* mais fáceis, possivelmente necessitando apenas de uma recompilação do código e alguns ajustes nos arquivos de cabeçalho da aplicação. Entretanto, na versão 12 houve uma reformulação do motor de cálculos do sistema, para que se adaptasse aos novos sistemas operacionais de 64 bits, que utilizam mais eficazmente o potencial dos processadores de núcleo duplo. Como o C++ é uma linguagem muito sensível no que diz respeito ao gerenciamento da memória, é muito provável que boa parte do código dos *plug-ins* desenvolvidos para o ArchiCAD tenha que ser novamente refeita. Essa situação é consequência da possibilidade de acesso em um nível mais baixo do que o conseguido com os *scripts* GDL, que podem ser executados em várias versões posteriores à que foram criados.

4.4 Acesso de modelos no formato IFC-SPF

Neste experimento foi testado o acesso a dados de modelos no formato SPF (STEP *Physical File*), descritos através do esquema IFC. Foram realizadas três tentativas utilizando duas ferramentas diferentes, com algoritmos em duas linguagens de programação (C++ e Java). Erros durante a compilação de códigos-fonte fornecidos com as ferramentas inviabilizaram a criação de ferramentas para ilustrar este modo de acesso. Nos itens seguintes as etapas do experimento são descritas.

4.4.1 Definição do problema

Acessar um modelo proprietário implica na aceitação da limitação do seu escopo e na dependência de um fabricante específico, como exposto nos experimentos anteriores. Formatos neutros de dados como as IFC, por outro lado, são criados com o propósito de facilitar a transferência de informações do modelo do edifício por diferentes fases do seu desenvolvimento. Em situações onde se pretenda acessar dados em um escopo mais amplo – isto é, mais fases do ciclo de vida da edificação – ou nas que seja mais interessante manter a aplicação mais adaptável a um contexto de múltiplas aplicações, acessar dados de modelos neutros torna-se mais aconselhável.

4.4.2 Desenvolvimento do experimento

A forma recomendada para acessar os dados de arquivos IFC em arquivos físicos (*STEP Physical File*) é utilizar uma ferramenta *IFC Toolbox*. A IAI sugere várias *IFC Toolboxes* em sua página internacional (IAI, 2008a). Essa ferramentas compilam um arquivo contendo a definição em EXPRESS das IFC e geram uma biblioteca contendo uma estrutura hierárquica de classes em C++, correspondentes às entidades do arquivo de definição. Essa operação é regida por especificações da própria norma ISO 10303, parte 22 – Standard Data Access Interface, conhecida como SDAI, que define um modelo para a criação de APIs (Application Programming Interfaces) para acesso a dados STEP (ISIKDAG *et al.*, 2007). Além de classes para cada uma das entidades, são criadas outras que gerenciam o acesso aos arquivos SPF (ou seja, os modelos de edifícios) que serão lidos. Por questões apenas de diferenciação, convencionou-se que

os arquivos SPF contendo modelos descritos de acordo com as IFC possuam a extensão ".ifc". O processo de criação dessa biblioteca de classes que intermedia o acesso aos modelos de dados é conhecido como *binding*.

Depois de criada, a biblioteca é utilizada pela aplicação em desenvolvimento para acessar os dados dos modelos de edifícios. As funcionalidades oferecidas pela biblioteca criada dependem da *IFC Toolbox* utilizada. *Toolboxes* genéricas criam apenas as classes equivalentes às entidades e tipos correspondentes aos atributos dessas entidades. Associar as diferentes entidades (por exemplo, todas as portas e janelas que fazem parte de uma parede) ou gerar representações e análises a partir delas fica a cargo do programador, como demonstrado no trabalho de Treeck e outros autores (2003). *Toolboxes* de programação em alto nível, além das classes e tipos, criam rotinas para associar hierarquicamente as entidades distribuídas pelo arquivo SPF, podendo a partir de então fornecer coleções de entidades. Essas coleções reduzem o trabalho de programação necessário para se obter informações a partir dos modelos de edifícios. *Toolboxes* de programação em alto nível podem, inclusive, conter rotinas para representação tridimensional das entidades do modelo.

Neste experimento foram testadas duas *Toolboxes*. A IFC Engine DLL, do instituto holandês TNO (TNO, 2008c) e a JSDAI EXPRESS API da alemã LKSoftWare (LKSOFTWARE, 2008b). A JSDAI não é uma *IFC Toolbox*, em seu sentido estrito. Na verdade o seu propósito é permitir a operação de *binding* a partir de qualquer esquema de dados descrito em EXPRESS. Sendo essa justamente a natureza das IFC, pode-se supor que a leitura de um modelo IFC a partir de uma EXPRESS *toolbox*, embora não se ofereça algumas funcionalidades específicas incluídas nas IFC *toolboxes*, em especial a geração de representações tridimensionais, deva resultar no mesmo conjunto de dados básicos (*raw data*). A escolha dessas duas ferramentas foi baseada em critérios de viabilidade e também na possibilidade de demonstrar duas formas diferentes de acesso aos modelos SPF. O quadro 4.01 mostra uma comparação entre as duas ferramentas *binding*.

Aspecto	IFC Engine DLL	JSDAI EXPRESS API
Proposta original	Pesquisa científica	Comercial
Linguagem de programação	C++	Java
Nível de programação	Alto (contém inclusive rotinas para modelagem tridimensional)	Baixo (apenas identifica as instâncias das entidades EXPRESS)
Compilação do esquema	Apenas versões das IFC	Qualquer esquema EXPRESS validável

Quadro 4.01: diferenças entre IFC Engine DLL e JSDAI EXPRESS API

Em comum, ambas as ferramentas tem o fato de serem gratuitas para propósitos não comerciais. Durante a preparação do experimento outras *IFC Toolboxes* sugeridas pela IAI foram consideradas, mas a maioria possuía versões apenas comerciais.

O processo de desenvolvimento de ferramentas de acesso seguiu as instruções contidas nas documentações e códigos de exemplo de ambas as ferramentas. No caso da IFC Engine DLL, foram adotadas duas abordagens: a primeira foi a programação em Java utilizando métodos nativos de acesso às classes C++, e segunda foi a programação apenas em C++. Em ambos os casos, optou-se por iniciar o desenvolvimento a partir da operação de leitura de um modelo IFC simplificado, e a partir daí construir uma ferramenta simples para ilustrar o acesso ao modelo. Os resultados dos experimentos com as duas *IFC Toolboxes* utilizadas é descrito nos itens a seguir.

IFC Engine DLL

A TNO disponibiliza exemplos de códigos-fonte para a operação de leitura de modelos a partir de várias linguagens de programação (TNO, 2008b). Não foram acrescentadas modificações a estes exemplos nesta fase do experimento. Os códigos foram baixados e compilados seguindo as instruções para cada linguagem utilizada (C++ e Java), e a compilação foi bem sucedida nas duas linguagens. A partir da aplicação compilada, foi possível acessar os dados do modelo que acompanha o código exemplo, e também de modelos criados no ArchiCAD e exportados para o formato IFC. Entretanto, esse acesso ocorre em um nível muito baixo. Como descrito na subseção 3.6.2, a estrutura de dados do modelo IFC é composta por uma série de entidades que decompõem os elementos construtivos até os seus primitivos geométricos. Para permitir um acesso em nível mais alto, a IFC Engine DLL tem rotinas que calculam a

superfície das entidades EXPRESS que representam sólidos geométricos e fornecem como parâmetros derivados as dimensões dos elementos (TNO, 2008a). Esse processo de "recomposição" do modelo é executado pelas classes `initializeModelling` e `initializeModellingInstance`. O uso dessas classes, porém, gerou erros de execução tanto na programação em Java como na em C++. Esses erros aconteceram durante a execução dos próprios exemplos fornecidos pela TNO, o que sugere que sejam necessárias configurações adicionais na compilação. A documentação existente não menciona essa situação e não foi possível contornar o erro em tempo hábil para o desenvolvimento de uma ferramenta preliminar. Uma possível solução seria utilizar apenas as classes de acesso aos dados em baixo nível, e produzir uma versão própria da classe `initializeModelling`.

JSDAI

Em relação à IFC Engine DLL, a ferramenta JSDAI oferece mais documentação, inclusive tutoriais em vídeo mostrando a criação de aplicações passo a passo. A ferramenta também está disponível na versão de um *workbench* para o ambiente de programação Eclipse, inclusive com um módulo para geração de esquemas via EXPRESS-G, a versão gráfica do EXPRESS (LKSOFTWARE, 2008b). Ao contrário da IFC Engine DLL, porém, a JSDAI não foi criada objetivando o acesso a arquivos IFC, e sim quaisquer arquivos SPF. Por isso, antes de desenvolver a aplicação em si, é necessário compilar o esquema EXPRESS que dará origem às classes que serão utilizadas no processo de *binding*. A compilação de esquemas EXPRESS em JSDAI é feita no módulo EXPRESS Compiler, também disponível na versão para Eclipse (LKSOFTWARE, 2008a).

A compilação do esquema IFC 2X3 no EXPRESS Compiler gerou um erro de validade, e a biblioteca de classes não foi gerada. Logo de início cogitou-se a possibilidade do erro ser causado pela JSDAI, e não pelo esquema IFC 2X3. O esquema era a versão estável mais recente disponibilizada pela IAI, e antes de ser disponibilizado havia sido certificado por várias ferramentas próprias para verificação da coerência de esquemas EXPRESS. Assim como no caso da TNO, a LKSofWare não foram encontradas soluções na documentação e nos fóruns de usuários da ferramenta. Para identificar o local exato do erro, uma vez que o EXPRESS Compiler

informava apenas uma falha de validade no esquema, eliminou-se do arquivo do esquema todas as entidades, mantendo-se apenas o cabeçalho e as declarações de tipo (*TYPE*). Feito isso, a compilação foi realizada sem erros e a biblioteca Java foi gerada, nesse caso contendo apenas funções de acesso genéricas e a declaração dos tipos equivalentes aos *TYPE* da EXPRESS.

Demonstrada a possibilidade de compilação, procedeu-se uma metódica tarefa de adição gradual de grupos de entidades (*ENTITY*) relacionadas, e compilações sucessivas. O erro inicial não reapareceu até que fossem inseridas no esquema as duas declarações *RULE* do final do esquema IFC 2X3: `IfcRepresentationContextSameWCS` e `IfcSingleProjectInstance`. Ficou claro que a EXPRESS Compiler da JSDAI não estava aceitando as duas declarações, embora a documentação da ferramenta afirme que esse tipo de estrutura pode ser interpretada na compilação. Do ponto de vista do desenvolvimento de algoritmos, o experimento poderia ter prosseguido sem as duas declarações *RULE*, mas, mesmo que uma aplicação fosse gerada, ela teria partido de um modelo de dados que não representaria fielmente o esquema IFC, que era o princípio do experimento. Embora não tenha sido encontrada explicação para o erro, pode-se argumentar que ele tenha sido causado pela característica essencialmente genérica da ferramenta SDAI. Nesse mesmo sentido, Nour e Beucke observam que não há linguagem de programação na qual as estruturas de dados EXPRESS possam ser mapeadas perfeitamente. As definições *RULE*, assim como os atributos derivados e inversos, tem sido um problema constante para as pesquisas no tema (NOUR e BEUCKE, 2008).

4.4.3 Discussão

É impróprio falar de vantagens verificadas por este experimento, pois as três abordagens empregadas para acessar o modelo IFC foram mal sucedidas. Isso não refuta de forma alguma o potencial de acessar o modelo de edifício por este método, apenas indica que mais tempo e preparação serão necessários em experiências futuras. Além disso, houve ainda outro experimento com o acesso ao modelo IFC, mas no formato ifcXML, que apresentou resultados melhores e é descrito na subseção seguinte.

Deve-se salientar que de modo geral a documentação encontrada foi muito menos explicativa que a utilizada nos experimentos de acesso ao modelo proprietário. Foi difícil formar um quadro mais amplo do acesso ao modelo apenas com as informações disponíveis nas páginas das duas ferramentas utilizadas, e a página da IAI nesse caso ajudou muito pouco. Também não é possível generalizar as conclusões desse experimento para as ferramentas comerciais que não puderam ser testadas. Nour, em sua tese de doutorado, relata experiências bem sucedidas de acesso a modelos IFC utilizando a *toolbox* da EPM Technologies (NOUR, 2006). Em trabalho mais recente, o autor aponta uma alternativa para a compilação do esquema IFC usando o Java Compiler Compiler – JCC (NOUR e BEUCKE, 2008). Este é um caminho promissor para desenvolvimentos futuros.

4.5 Acesso de modelos no formato ifcXML

O último dos experimentos realizados neste trabalho seguiu uma sequência similar ao anterior, porém com melhores resultados. Apesar da documentação para orientar o desenvolvimento de aplicações para acessar arquivos ifcXML ser igualmente escassa, a tecnologia XML é notadamente mais simples. Além disso, documentações e ferramentas genéricas, que podem ser adaptadas para o esquema ifcXML são amplamente disponíveis. Um modelo em formato ifcXML tem basicamente os mesmos benefícios de um modelo em formato IFC (SPF), porém descrito de uma maneira diferente. Uma rápida introdução sobre o formato é apresentada na subseção seguinte.

4.5.1 ifcXML

Extensible Markup Language, ou XML, é uma linguagem de metadados recomendada pelo World Wide Web Consortium (W3C) como padrão para troca de informações na Web (W3C, 2008b). Trata-se de uma linguagem textual para descrição de informações, que permite reunir em um só local tanto os dados como o significado semântico – os metadados. Ao contrário da EXPRESS, que foi criada com o propósito de representar modelos de produtos, a XML é essencialmente genérica, e pode ser utilizada para descrever qualquer tipo de informação. Este alto nível de abstração

possibilitou a disseminação da linguagem, mas é um obstáculo à garantia de consistência da informação.

Como não tem descritores e estruturas fixas, a XML pode representar qualquer informação de infinitas maneiras, e qualquer uma delas será válida, desde que sigam umas poucas regras de sintaxe. Por exemplo, elementos construtivos podem ser chamados `<segmento_de_parede>` ou `<parede>`, e podem conter descritores diferentes para representar seus atributos. E ainda, a posição de uma parede pode ser descrita em relação a uma origem fixa, ou então em relação às outras paredes, e pode seguir um sistema cartesiano ou polar. Em resumo, há infinitas formas de descrever corretamente um edifício, o que para humanos torna-se uma questão de semântica rapidamente resolvida mas para computadores pode exigir precisos mapeamentos de dados. Para resolver situações como esta, foi desenvolvido XML *Schema Definition*, ou XSD. XSD é em essência uma linguagem de meta-metadados. Ela permite descrever as estruturas que descrevem estruturas de dados (assim como a EXPRESS). Um esquema XSD define tanto os descritores que podem ser utilizados em um arquivo, como as relações possíveis e obrigatórias entre eles. Por exemplo, pode-se definir que um descritor `<janela>` tenha que ser obrigatoriamente associado a um descritor `<wall>`. Ao se criar arquivos XML a partir de esquemas XSD, é possível validar o conteúdo da informação utilizando o esquema como referência, e determinar a consistência da informação.

A ifcXML é justamente um esquema XSD para formatação de arquivos XML de acordo com as estruturas das IFC. No processo de criação do esquema XSD, que é conduzido pela IAI e segue a metodologia definida na STEP, as entidades e relacionamentos do esquema EXPRESS IFC são mapeados e dão origem a descritores XML (NISBET e LIEBICH, 2007), que são então utilizados para representar os elementos dos modelos de edifícios. A primeira versão da ifcXML foi lançada em 2001, e a partir de então, para cada versão do esquema IFC, há uma versão ifcXML correspondente, mapeada automaticamente.

4.5.2 Definição do problema

O objetivo de usar a ifcXML é permitir a criação de aplicações para acessar modelos sem que se exija dos programadores o domínio de esquemas EXPRESS, já que há pouca documentação e ferramentas para auxiliar a programação. A própria IAI reconhece que a linguagem XML tem uma inserção muito mais ampla do que seria possível com a EXPRESS, mesmo se considerado apenas o setor da indústria da construção. Por isso, definiu que a utilização das duas formas de IFC é complementar, sendo a ifcXML mais recomendada para transferir porções da informação do modelo, enquanto os arquivos SPF, mais compactos, são utilizados para transmitir o modelo completo do edifício (NISBET e LIEBICH, 2007). Aranda-Mena e Wakefield vão ainda além, afirmando que a ifcXML, por ser baseada em uma tecnologia da Web e naturalmente associável aos recentes desenvolvimentos da web semântica, como a linguagem de ontologias OWL, provavelmente dará origem a uma nova geração de padrões para interoperabilidade na construção (ARANDA-MENA e WAKEFIELD, 2006).

4.5.3 Desenvolvimento do experimento

Arquivos XML podem ser lidos como strings de texto, mas essa abordagem demanda um grande esforço de programação. Uma maneira mais eficiente é utilizar APIs próprias para a leitura do formato XML, que identificam automaticamente os descritores, atributos e valores. Utilizando essas ferramentas, a conexão com um arquivo XML pode ser feita via *parsing* ou via *binding*. Neste experimento, as duas alternativas foram testadas.

Parsing

O processo de leitura por *parsing* pode ser descrito como uma varredura sequencial da estrutura de nós do arquivo XML. A ferramenta percorre os nós, identificando cada descritor, seus atributos e os valores, além de associá-los hierarquicamente. Aplicações que utilizam *parsing* podem tanto executar uma varredura completa de todos os nós, armazenando-os em estruturas temporárias para um acesso mais rápido, ou executar várias varreduras, sempre que uma informação precisa ser lida. O mesmo se aplica ao processo de gravação das informações.

As ferramentas utilizadas no experimento de *parsing* foram a Simple API for XML – SAX e a Document Object Model – DOM. A SAX foi a primeira API para XML largamente adotada, inicialmente desenvolvida apenas para Java (SAXPROJECT, 2008). A DOM é definida pelo W3C como uma interface independente de linguagem e plataforma que permite programas e *scripts* acessarem e atualizaram dinamicamente o conteúdo e a estrutura dos documentos XML (W3C, 2008a). Tanto a SAX como a DOM são disponíveis no pacote de programação Java (JDK), que foi a linguagem de programação utilizada. Ambas permitem criar aplicações para acessar arquivos ifcXML com umas poucas linhas de código. As especificidades dos arquivos ifcXML são irrelevantes para testar o acesso inicial, que pôde ser realizado com os próprios exemplos contidos nas páginas das duas aplicações a nas documentações sobre Java consultadas (DÉCIO, 2000; SCHILDT, 2002; VELOSO, 2007; JANDL, 2008).

De todos os métodos descritos neste trabalho para acessar dados dos modelos de edifícios, este foi sem dúvida o mais simples. As funcionalidades para validar o conteúdo, entretanto, são limitadas. A SAX é mais aconselhável para situações de leitura de pequenas sequências de dados, como arquivos de configuração de parâmetros de interface, ou criação de páginas da Web. Exceto pelos métodos que permitem ler e gravar o arquivo XML, que evitam a necessidade de programar rotinas para manusear e separar *strings* (*tokenizers*), a SAX não oferece funcionalidades adicionais. Trabalhar com longas e interrelacionadas sequências de nós, que são o caso dos arquivos ifcXML, exige a programação de rotinas complementares para estruturar os dados e verificar a consistência da informação. A rotina de estruturação dos dados consistiria em associar os descritores do arquivo ifcXML a tabelas de dados ou classes que descrevessem as entidades IFC. Os atributos dos descritores seriam então convertidos em registros das tabelas ou atributos das instâncias das classes. Verificar a consistência da informação exigiria uma rotina para comparar os nós e as relações entre eles com as definições do esquema XSD da ifcXML. Ou uma sequência de comparações seria programada diretamente no código – o que tornaria difícil atualizar o programa quando fosse lançada uma nova versão da IFC, ou então uma classe de comparação poderia ser criada, para a qual seriam passadas as definições do arquivo contendo o esquema. Qualquer que fosse a opção escolhida para a rotina de

verificação da consistência, exigiria um árduo trabalho de programação, que pouco se aproveita da estruturação semântica já inserida no esquema XSD da ifcXML. Considerando isso, o experimento com o acesso via SAX foi considerado concluído.

O passo seguinte foi introduzir a DOM, que é uma API mais sofisticada do que a SAX. Após ler um documento XML, a DOM cria uma estrutura de objetos representando os seus nós, com métodos "*get*" para permitir o acesso aos seus atributos. Uma vez lido o arquivo, o trabalho é realizado sobre essa estrutura de objetos, chamada `documentElement`, eliminando as múltiplas varreduras da SAX. Outra vantagem é o relacionamento hierárquico criado pela DOM: instâncias dos nós relacionados são acessadas diretamente, a partir dos métodos `getChildren` e `getParent`. Desse modo, as entidades aninhadas do arquivo ifcXML são estruturadas automaticamente, dispensando a rotina que teria que ser criada com a SAX. Ainda haveria, porém, a necessidade de criar uma rotina para verificação da consistência da informação contida no arquivo, que dependeria do mapeamento entre um `documentElement` extraído a partir do arquivo XSD e outro `documentElement` contendo a informação do modelo.

Os métodos disponibilizados pela SAX e pela DOM são essencialmente abstratos, o que é útil para acessar pequenos arquivos e para manter o código do programa flexível para suportar a evolução dos esquemas IFC. Porém para trabalhar com estruturas mais complexas, muita programação adicional é exigida. Para ilustrar essa dificuldade, considere-se o arquivo ifcXML representando um modelo criado no ArchiCAD, apresentado no Apêndice F. Após o *parsing*, a DOM retorna um objeto `documentElement` contendo a estrutura hierárquica de nós do arquivo ifcXML. O `documentElement` é composto por `elements`, que representam nós, atributos e valores do arquivo lido. Cada entidade é representada por uma estrutura abstrata, então para identificar o tipo delas, é preciso obter o atributo `name` dos objetos `element`. Pode-se também obter coleções de entidades de um mesmo tipo, passando para o método `getElementsByTagName()`. Entidades IFC mais simples podem então ser acessadas facilmente. Por exemplo, para se obter o nome do primeiro pavimento do modelo, utiliza-se a expressão:

```
String name =
documentElement.getElementsByTagName(“IfcBuildingStorey”).
      item(0).getAttribute(“Name”);
```

Porém, as principais entidades IFC, que representam elementos construtivos, são estruturas de várias entidades aninhadas. Para obter as coordenadas de um ponto da forma geométrica que representa um elemento construtivo, a expressão seria a seguinte:

```
Real x =
documentElement.getElementsByTagName(“IfcShapeRepresentation”).
      item(0).getAttribute(“Items”).getElementsByTagName
      (“IfcPolyline”).item(0).getAttribute(“Points”).
      getElementsByTagName(“IfcCartesianPoint”).item(0).
      getAttribute(“Coordinates”).getElementsByTagName
      (“IfcLengthMeasure”).item(0).getNodeValue();
```

Expressões abstratas, enormes e não recomendadas como esta dificultam a documentação do programa e praticamente inviabilizam modificações. Para evitá-las, poderiam ser criadas várias rotinas que simplificassem a criação de expressões. Quanto mais se quiser simplificar a expressão, mais trabalho de programação será necessário, até que se chegue a algo do tipo:

```
Real x = model.getEntity(“IfcShapeRepresentation”, 0).
getValue(“Item”, 0).getValue(“Points”, 0).
getValue(“Coordinates”, 0).getValue();
```

Note-se que os métodos foram condensados, mas ainda permaneceram abstratos. Abstração nesse nível é um problema para modelos de dados bem definidos e interoperáveis como as IFC. É preciso garantir que toda a informação processada e produzida pela aplicação seja validável – ou seja, que esteja de acordo com o esquema ifcXML. Mantendo a abstração de dados mostrada, nada impediria a geração de um arquivo ifcXML com uma entidade chamada `minha_parede`, caso o programador passasse esse valor para a propriedade *name* de um objeto `node` da DOM. A próxima etapa, então, seria criar classes concretas para refletir as estruturas das entidades IFC e limitar a possibilidade de definições impróprias durante a programação. A classe `ifcShapeRepresentation()` poderia representar a entidade IFC de mesmo nome, evitando que um erro de digitação produzisse uma entidade inválida. Além disso, ambientes de programação – como o Eclipse – possuem ferramentas para auxiliar o processo de geração e manutenção dos códigos dos programas. Uma estrutura bem

definida de classes é uma ótima maneira de reduzir erros durante a programação e facilitar o entendimento da estrutura de dados para os programadores. Em última instância, códigos contendo erros de digitação nos nomes das classes ou chamadas de métodos em situações inadequadas sequer seriam compilados e, portanto, não existiria a possibilidade de gerar arquivos ifcXML não validáveis.

Isso exigiria a definição de mais de 620 classes, uma para cada entidade IFC, e milhares de variáveis e métodos de acesso encapsulados. Mesmo que essa tarefa fosse enfrentada e concluída, a atualização dessa estrutura sempre que o esquema IFC fosse modificado seria algo desanimador. Há, porém, ferramentas para automatizar esse processo – as APIs para mapeamento de modelos de dados, ou *binding*.

Binding

Arquivos ifcXML podem ser acessados por *binding*, o mesmo processo utilizado pelas ferramentas IFC Toolbox da seção 4.4. No lugar de compilar o esquema IFC original, em EXPRESS, compila-se o esquema XSD derivado do original, fornecido pela IAI. Neste experimento foi utilizada a ferramenta XMLBeans, da Apache (APACHE, 2008). A XMLBeans é gratuita e tem uma rica documentação fornecida pelo desenvolvedor, incluindo tutoriais passo a passo.

O processo de *binding* começa com a compilação do esquema XSD da ifcXML através da aplicação Scomp, fornecida com o pacote XMLBeans. O esquema ifcXML é composto por dois arquivos: ex.xsd e IFC2X3.xsd. O primeiro arquivo contém definições padrão de tipos simples de dados e relacionamentos entre entidades EXPRESS, segundo a ISO 10303. O segundo arquivo contém as definições dos tipos complexos de dados e entidades IFC, transpostas para o formato XSD. Ambos os arquivos podem ser obtidos gratuitamente no site do grupo de modelagem da IAI (IAI, 2008c). Para funcionar, a Scomp necessita do pacote Java Development Kit (JDK), e da configuração da variável de ambiente PATH, do Windows.

Como o esquema XSD da ifcXML é muito extenso (21.505 linhas), tentar compilá-lo com a configuração padrão da Scomp resulta em um erro de memória (*System is out of resources*). Então é necessário alocar mais memória para a aplicação,

usando o parâmetro `–MX 1024M` na linha de comando. Com esse pequeno ajuste o erro é resolvido, e a compilação pode ser completada, gerando duas bibliotecas de classes Java:

```
orgStandard10303Part28Version2XmlschemaCommon.iso
org.iaiTech.ifcXML.ifc2X3.xfinal
```

Cada biblioteca contém as classes e tipos correspondentes às entidades definidas nos dois arquivos do esquema. Ambas precisam ser importadas pela aplicação Java para o acesso a modelos ifcXML. A partir da importação, classes correspondentes às entidades IFC e classes mais genéricas para gerenciar a leitura e gravação de arquivos XML ficam disponíveis para utilização no código. Como exposto na subseção anterior, a estrutura bem definida de classes e métodos conduz o programador no acesso à informação IFC, e protege a aplicação de erros de digitação e chamadas de métodos em situações inadequadas. Se erros forem cometidos, a compilação informa a situação e não executa a aplicação. Essas vantagens podem ser ampliadas usando um ambiente de programação, como o Eclipse, que contém ferramentas de auxílio à programação sensíveis ao contexto. A outra grande vantagem é a possibilidade de verificação da consistência da informação contida no arquivo a ser lido, ou seja, a validação do XML de acordo com a estrutura do esquema XSD, a partir de um método gerado automaticamente.

Há duas formas de *binding*: *late* e *early*. No *late binding*, apenas as classes genéricas de acesso ao arquivo e as que aceitam parâmetros abstratos são utilizadas. Por exemplo, a classe `getEntity()` pode ser utilizada para acessar qualquer entidade do arquivo. Nesse caso, os atributos tornam-se também abstratos, e precisam ser identificados através de constantes literais. Por exemplo, com o método `entity.getAttribute(“id”)`. As desvantagens desse método são essencialmente as mesmas da utilização da SAX e da DOM, embora o processo de validação seja mais simplificado. O *Late binding* é recomendado para situações onde o esquema dos arquivos a serem lidos pela aplicação não é definido. Gerar aplicações com algo grau de abstração pode torná-las mais flexíveis aos modelos de dados, porém o trabalho de modelagem de processos e programação de rotinas é muito mais complexo. Além disso, para o caso dos modelos de edifícios há esquemas previamente definidos, e faz

pouco sentido criar aplicações excessivamente abstratas. Desse modo, o experimento realizado neste trabalho adotou a abordagem *early binding*, pela qual identifica-se a estrutura de dados diretamente no código da aplicação, de modo mais concreto. Em vez de `getEntity()`, utiliza-se a classe correspondente à entidade ifcXML, como `getWall()` ou `getBeam()` por exemplo. Os métodos, em consequência, também são identificados, como `entity.getID()` no lugar do exemplo mostrado anteriormente.

Uma pequena aplicação foi preparada para ler modelos e testar as bibliotecas geradas pela compilação do esquema. A aplicação lê um arquivo ifcXML, informa se a operação de leitura foi bem sucedida, computa e informa o número de entidades IFC, e mostra como exemplos o acesso aos dados descritos em duas entidades: `IfcSite` e `ifcWallStandardCase`. O código completo da aplicação é apresentado no Apêndice G.

Nos primeiros testes a aplicação não conseguiu abrir arquivos ifcXML gerados no ArchiCAD 11 e no ArchiCAD 12. A classe responsável pela leitura, `Iso1030328Document`, retornava erro de má-formação. Isso indicava problemas de formatação ou caracteres incorretos. De fato, foi observado que os arquivos ifcXML gerados pelo ArchiCAD indicavam a codificação Unicode UTF-8 no atributo dos seus cabeçalhos. Por isso, qualquer elemento do modelo que tenha sido nomeado utilizando caracteres considerados internacionais provocava o erro de formação do arquivo ifcXML. Como não foi possível identificar uma opção para modificar a codificação durante o processo de exportação do ArchiCAD, foi desenvolvida uma pequena rotina para substituir o atributo no arquivo para “ISO-8859-1”, a codificação que suporta caracteres dos alfabetos latinos. Nos testes seguintes a classe que lia os arquivos considerou-os bem formados, e o experimento pode prosseguir.

Algumas divergências entre os *namespaces* do esquema XSD e dos arquivos gerados pelo ArchiCAD exigiram uma sequência adicional de operações na abertura dos arquivos. *Namespaces* diferentes dos definidos no esquema não estavam sendo reconhecidos, por mais que se passasse opções para a classe de leitura dos arquivos. Após uma série de tentativas infrutíferas, decidiu-se por simplesmente substituir os atributos do nó principal dos arquivos lidos pelas versões reconhecidas pelo esquema

original (NISBET e LIEBICH, 2007), antes da leitura das entidades. Por exemplo, o *namespace* definido pelo URI `urn:oid:1.0.10303.28.2.1.3`, do arquivo gerado pelo ArchiCAD foi substituído por

`urn:iso.org:standard:10303:part(28):version(2):xmlschema:common`

Desse modo, a leitura dos arquivos foi concluída e pôde-se obter as entidades IFC do modelo. Porém, uma revisão minuciosa sobre as causas desses conflitos, que não pode ser realizada no decorrer desse trabalho, deve ser conduzida para identificar as causas desse comportamento. Em teoria, *namespaces*, que são apenas desambiguadores para o caso de entidades de origens diferentes que tenham o mesmo nome, não deveriam produzir um erro de consistência. Considerando isso, o problema encontrado deve ser abordado como uma questão de ajuste das opções da compilação do esquema na XMLBeans.

Depois que o arquivo é lido, uma estrutura semelhante à criada pela DOM é disponibilizada para o acesso aos dados do modelo. Nesse momento, porém, foi observado que a facilidade para o acesso aos dados por meio de classes e métodos completos, correspondendo às entidades IFC não era possível. O esquema IFC agrupa todas as entidades do modelo do edifício dentro da entidade chamada UOS (*Unit of Serialization*). Em outros ramos da indústria, essa entidade reúne informação suficiente para a produção de uma unidade seriada. Como podem ser produzidos vários tipos de unidades em série, pode haver várias entidades UOS nos esquemas. No caso da IFC, considera-se que o produto é apenas um, e portanto recomenda-se o uso de uma única entidade UOS (LIEBICH e WIX, 2000). Ao gerar a estrutura de classes a partir do esquema ifcXML, a XMLBeans agrupa todas as classes de entidades sob a classe `Uos()`. O agrupamento em si não é um problema, pois pode-se pensar na classe como se fosse o próprio modelo, a partir da qual toda a informação seria extraída a partir de métodos “get”. No entanto, a compilação não integrou métodos concretos para acessar as diferentes entidades a partir da classe `Uos()`. Não há o método `getIfcSite`, por exemplo. Para acessar as entidades, há apenas um método abstrato chamado `getEntityArray`, que gera uma matriz de objetos `entity` (uma interface abstrata para as entidades). Para discernir entre as diferentes entidades da matriz, um

processo muito semelhante ao mencionado para o acesso via DOM seria necessário. Cada uma das entidades da matriz passaria por uma série de condicionais, até que o seu tipo, obtido pelo método `getClass()` fosse determinado.

Neste experimento, as vantagens do processo *binding* sobre o *parsing* começam a ser percebidas apenas após a conclusão desse processo de identificação dos tipos de cada entidade. A partir daí, as classes correspondentes às entidades IFC tornam-se extremamente úteis. No código da aplicação, mostrado no Apêndice G é mostrada a identificação da entidade `IfcSite`, cujo valor é atribuído para uma instância da classe correspondente, `IfcSite()`, gerada pela XMLBeans. A classe `IfcSite()` encapsula métodos para acesso dos seus atributos, então para acessar o grau da longitude do ponto de referência do terreno do projeto, a expressão seria:

```
long degree =
site.getRefLatitude().getLongWrapperArray(0).getLongValue();
```

A latitude completa é um tipo de dado composto, definido pela classe `compoundPlaneAngleMeasure()`, que contém uma matriz de três números *long* para armazenar grau, minuto e segundo. Então, para obter os próximos valores, o parâmetro passado para o método `getLongWrapperArray` seria mudado para 1 e depois para 2.

Como se pode ver, os nomes dos métodos refletem a estrutura da entidade, definida na EXPRESS, convertida para esquema XSD, e posteriormente compilada pela XMLBeans. Não seria possível acessar ou atribuir ao elemento `site`, que é do tipo `IfcSite`, nada que não fosse determinado pelo esquema, garantindo a consistência da informação e facilitando o trabalho de programação. A última questão a ser avaliada era a possibilidade de acesso aos dados de entidades aninhadas. No código apresentado no Apêndice G foi identificada uma entidade `ifcWallStandardCase`. No Apêndice F é mostrado o arquivo ifcXML utilizado no teste, e na página seguinte apenas a entidade lida, para ilustração. A `ifcWallStandardCase` é formada por vários atributos, entre eles o `Representation`, que contém outras entidades IFC aninhadas, que são os elementos que constituem a representação formal da parede.

```
<IfcWallStandardCase id="i1714">
      <GlobalId>1WfU$nJCrCZO_HrHunLtl5</GlobalId>
      <OwnerHistory>
       <IfcOwnerHistory xsi:nil="true" ref="i1568"/>
      </OwnerHistory>
  <Name>SW - 002</Name>
  <ObjectPlacement>
    <IfcLocalPlacement xsi:nil="true" ref="i1711"/>
  </ObjectPlacement>
  <Representation>
    <IfcProductDefinitionShape id="i1793">
      <Representations ex:cType="list">
        <IfcShapeRepresentation pos="0" xsi:nil="true" ref="i1753"/>
        <IfcShapeRepresentation pos="1" xsi:nil="true" ref="i1786"/>
      </Representations>
    </IfcProductDefinitionShape>
  </Representation>
  <Tag>3B09AD2E-868F-48F9-A4-80-F4F953F6B0E1</Tag>
</IfcWallStandardCase>
```

Enquanto as entidades aninhadas são descritas sequencialmente, como o caso da `ifcProductDefinitionShape`, é possível acessá-las como o resultado de um método "get" da classe hierarquicamente acima. Por exemplo, para acessar a primeira entidade `ifcShapeRepresentation` aninhada, a expressão seria

```
wall.getRepresentation().getIfcProductRepresentation().
getRepresentations().getIfcRepresentationArray(0);
```

Entretanto, o objeto retornado por essa expressão, uma `IfcShapeRepresentation`, não contém nenhum parâmetro. Isso porque trata-se apenas de uma referência ao objeto original, que está descrito em outra parte do arquivo ifcXML. Essa propriedade é utilizada pelos esquemas IFC para evitar aninhamentos muito complexos, e também para permitir que uma mesma entidade seja utilizada por várias outras sem que seja necessário reescrevê-la (BERARD e SHULZ, 2008). O código "i1753" do parâmetro `ref` da entidade `IfcShapeRepresentation` é chamado g*lobal ID* e é utilizado para localizar a entidade referenciada. Essa operação não é automatizada pela XMLBeans, e cabe ao programador realizar a busca percorrendo todas as entidades do arquivo em busca da que contenha o g*lobal ID* desejado. A expressão mostrada anterior acrescida do parâmetro `getRef()` retorna o *Global ID* a ser localizado. No código completo da aplicação de teste é mostrada uma operação para localizar o objeto referenciado.

Depois de localizada a entidade `ifcShapeRepresentation` contendo o g*lobal ID* desejado, pode-se acessar os seus componentes geométricos a partir do método `getItems()`, que resulta uma matriz de elementos de representação. No arquivo utilizado no teste, essa matriz continha apenas um elemento, o polígono (`ifcPolyline`) que no ArchiCAD daria origem à parede através de um processo de extrusão. A `ifcPolyline`, por sua vez, contém pontos cartesianos que são acessados pelo método `getPoints()`. Esse processo de identificação dos elementos e acesso dos seus atributos repete-se até que toda a informação a respeito do elemento construtivo tenha sido recolhida. A expressão que retornaria a abscissa do segundo ponto geométrico que forma a polilinha que representa a base da parede seria assim:

```
double x = polyline.getPoints().getIfcCartesianPointArray(1).
getCoordinates(). getIfcLengthMeasureArray(0).getDoubleValue();
```

O experimento permitiu concluir que embora a estruturação por classes correspondentes às entidades IFC seja uma poderosa ferramenta para auxiliar o programador, a distribuição das entidades pelo arquivo ifcXML requer rotinas para facilitar a recomposição da informação. Uma classe recursiva para automatizar o processo repetitivo de localização das entidades e dos seus atributos seria de grande ajuda.

4.5.4 Discussão

O acesso a modelos em formato ifcXML foi bem sucedido, e o experimento permitiu identificar várias possibilidades para o desenvolvimento de aplicações para um ambiente BIM. A grande popularidade do modelo de dados XML garantiu muitas opções de suporte técnico, um grande número de aplicações de apoio, e a fácil localização da documentação necessária. Ao contrário do experimento com acesso via IFC em formato SPF, os erros encontrados puderam ser resolvidos ou contornados. Embora não tenha sido gerada uma aplicação com funcionalidades mais avançadas, pode-se generalizar as observações e conclusões.

O uso da SAX e da DOM é recomendado apenas para aplicações mais simples, destinadas por exemplo a acessar porções bastante reduzidas dos modelos de edifícios, como, por exemplo, apenas identificar todos os materiais utilizados, mas não

o seu volume. O uso de uma ferramenta de *binding* como a XMLBeans resultou em uma estrutura de acesso aos dados mais adequada e flexível, e é a abordagem recomendada por este trabalho. Porém, as funcionalidades da ferramenta *binding* devem ser estudadas previamente, para evitar a necessidade de longas rotinas de identificação dos tipos das entidades, que tornam o código inflexível e de difícil manutenção.

5
Considerações finais

Como primeira contribuição deste trabalho deve-se citar a proposta de classificação do escopo da modelagem de produto na construção em quatro níveis de abstração, no sentido de possibilitar uma aproximação facilitada para um tema de implicações tão amplas. Em seguida, os experimentos realizados demonstraram que a criação de aplicações para acessar os modelos e gerar informações a partir deles é viável. Com exceção do acesso ao modelo no formato IFC SPF, todos os experimentos foram bem sucedidos no acesso e processamento de dados. A seguir é feita uma pequena revisão e considerações sobre cada método de acesso experimentado.

Os modelos proprietários possuem métodos de acesso mais fáceis, pois há um grande interesse das fabricantes de *softwares* em disseminar as suas aplicações e permitir a sua adaptação a contextos específicos, através da especialização das suas funcionalidades por *plug-ins* ou *scripts*. A documentação é mais acessível, e as funções de acesso são mais diretas, permitindo a criação de aplicações mais rapidamente. Os três tipos de acesso são a criação de *scripts*, a criação de *plug-ins* ou aplicações independentes.

Scripts foram utilizados no experimento descrito na seção 4.1 para gerar objetos representando elementos construtivos, com geração automática da documentação projetual. Esse tipo de acesso ao modelo mostrou-se bastante útil principalmente em situações onde as unidades de informação são razoavelmente autônomas, e o seu comportamento não precisa considerar a totalidade do modelo do edifício. Não foram identificados na linguagem utilizada (GDL) meios de aumentar o escopo da sensibilidade ao contexto para, por exemplo, permitir que o objeto reconheça outros objetos e suas relações a partir do *script* de comportamento.

Scripts também foram utilizados no experimento apresentado na seção 4.2, para associar as dimensões dos objetos paramétricos a informações complementares, gerando processos automáticos de quantificação. As observações realizadas a partir do experimento podem ser generalizadas para a quantificação de quaisquer características relacionadas aos elementos do projeto que possam ser convertidas em índices de consumo – por exemplo, custo por metro cúbico, horas de trabalho por metro quadrado, etc. *Scripts* de quantificação podem associar estes índices aos objetos representando elementos construtivos e também aos objetos representando elementos abstratos, como as áreas internas dos cômodos.

O método de acesso via *plug-in* foi testado na seção 4.3, com a criação de uma pequena aplicação para exportar objetos do ArchiCAD para o sistema Mestre de simulação física. A partir das bibliotecas fornecidas pela Graphisoft (Graphisoft API) é possível criar aplicações em C++ e utilizar funções para acesso ao modelo como se ele fosse um banco de dados relacional. Este método de acesso oferece mais flexibilidade e controle do acesso e da interface da aplicação do que o método *script*. A mesma biblioteca pode ser importada por uma aplicação independente, para acessar o modelo proprietário da Graphisoft a partir de uma interface independente da execução do ArchiCAD.

A desvantagem de utilizar métodos de acesso a modelos proprietários é a dependência das fabricante de *softwares* e a necessidade de constantes atualizações das aplicações complementares, sempre que uma nova versão da aplicação original é lançada. Modelos proprietários também exigem a criação de mapeamentos de dados, para cada aplicação a ser integrada durante o processo de desenvolvimento de edifícios.

Modelos neutros, como as IFC, tem como objetivo integrar diferentes aplicações através do uso de um único modelo de dados. Isso elimina a necessidade de criação de vários mapeamentos de dados entre aplicações diferentes. Os mapeamentos são realizados apenas uma vez, entre o modelo de dados e a nova aplicação sendo desenvolvida. Neste trabalho, foram testados dois métodos de acesso ao modelo IFC: via arquivo SPF e via arquivo ifcXML.

O experimento de acesso ao modelo IFC por arquivo SPF (seção 4.4) foi o único a não produzir resultados palpáveis. Há pouca documentação, e os manuais são confusos, com poucos exemplos. As duas ferramentas gratuitas utilizadas, TNO IFC Engine e LKSoftWare JSDAI, não funcionaram de acordo com o esperado, mesmo depois de seguidas as instruções das fabricantes. Novos testes com as configurações das ferramentas precisam ser realizados para identificar as causas dos problemas ocorridos no experimento. As várias ferramentas comerciais disponíveis que realizam o acesso a modelos IFC em formato SPF não foram testadas, e também são opções para aqueles que pretenderem realizar estudos complementares.

O acesso a modelos em formato ifcXML, apresentado na seção 4.5 foi bem sucedido, e por utilizar de uma tecnologia mais disseminada (XML), há mais documentações e tutoriais disponíveis. Foram identificadas três formas para acessar o modelo: lendo o arquivo como uma sequência de *strings*, o que exige um trabalho enorme de programação e não é recomendado; através de *parsing*, com ferramentas como a DOM ou a SAX, o que reduz o esforço de programação mas não auxilia o programador a identificar as entidades IFC; e através de *binding*, com a compilação das entidades do esquema ifcXML e geração de classes Java equivalentes. Essa foi a forma de acesso a modelos ifcXML que gerou melhores resultados e exigiu menor esforço de programação. É a abordagem recomendada para estudos subsequentes.

Em resumo, o acesso ao modelo foi bem sucedido, e o processamento destes dados demonstrou interessantes possibilidades para a geração de aplicações para a indústria da construção. Entretanto, o modelo neutro IFC, que há dez anos é anunciado como uma solução para a interoperabilidade, foi o que apresentou mais dificuldades para ser acessado. Há pouca documentação disponível, poucos exemplos e poucos fóruns onde as dúvidas possam ser dirimidas. Há poucos profissionais envolvidos tanto no desenvolvimento do modelo IFC em si como de ferramentas para acesso a ele. Não há versões gratuitas para as ferramentas de acesso mais citadas em trabalhos sobre o tema, e as opções gratuitas não corresponderam às expectativas.

Do ponto de vista do desenvolvedor de aplicações, o acesso aos modelos IFC deveria ser muito mais simples, considerando que se pretende que o padrão torne-se a

principal forma de transmissão de informações na indústria da construção. Uma das causas da baixa adoção do padrão pode ser a dificuldade de gerar rapidamente aplicações especializadas para o ambiente de interoperabilidade defendido pela IAI. Novas ferramentas devem ser desenvolvidas para facilitar e flexibilizar o acesso, fornecendo à comunidade de programadores um modo mais simples para realizar o trabalho.

5.1 Sugestão para trabalhos futuros: metacompilação de entidades

O processo de *binding* do esquema ifcXML provou ser de grande valia para o desenvolvimento de aplicações para acessar modelos de edifícios. As classes geradas pelo XMLBeans facilitam sensivelmente a extração dos dados, porém ainda mais útil para o programador seria uma ferramenta para acesso em um nível mais alto. Expressões resultantes do uso dessa ferramenta seriam, por exemplo:

```
real x = wall.getLength;

real vol = wall.getVolume;

real material =
wall.getVolume * wall.getComponentsByVolume(“cement”);
```

Os modelos proprietários de edifícios possuem esses métodos, porque tem estruturas de dados de nível mais alto, criadas para atender apenas as necessidades dos BIM CADs aos quais se destinam. Os modelos neutros, por outro lado, tem estruturas de dados mais abstratas, originando diferentes possibilidades de combinações dos dados e unidades de informação. Cada unidade de informação é decomposta em várias outras, até que sejam atingidos os tipos primitivos de dados, atômicos (ex. inteiro, *string*, real, booleano).

A estrutura de dados abstrata flexibiliza o acesso e permite a construção de vários mapeamentos entre o modelo IFC e o modelo de dados interno das aplicações em desenvolvimento. Porém lidar com essa abstração, como demonstrado na seção 4.5.3, pode dificultar muito o desenvolvimento de aplicações. O código torna-se muito complexo e de difícil atualização, propiciando o surgimento de erros.

Uma abordagem mais adequada seria associar os benefícios das duas propostas: manter a abstração de dados mas oferecer a possibilidade de geração de classes mais refinadas, para acesso em nível mais alto. Essas classes deveriam fornecer métodos que localizassem e associassem dados em diferentes partes do modelo – ou seja, diferentes atributos das várias entidades relacionadas em um modelo neutro. Esses dados seriam então processados para dar origem à informação em nível mais alto.

Uma solução para isso seria criar manualmente classes com funcionalidades mais finamente sintonizadas com as informações que são usualmente mais importantes para o desenvolvimento de aplicações para a indústria da construção. Entretanto, a questão do grande número de entidades e atributos a serem revistos, e a dificuldade de atualização das classes quando o esquema IFC fosse atualizado surge como séria desvantagem.

Uma outra solução, mais robusta, seria criar essas classes automaticamente, através de um processo de metacompilação, como faz o próprio XMLBeans, ou ainda o Java Compiler Compiler. Nesse caso, seriam necessários dois esquemas: o esquema original IFC, que contém as unidades de informação (entidades, atributos e relacionamentos) e um esquema complementar, para fornecer relacionamentos adicionais entre as entidades e atributos, para a geração dos métodos de acesso em nível mais alto.

Semântica é a questão chave para a criação automática dessas classes. O esquema original fornece a natureza dos elementos construtivos e seus componentes, e o esquema complementar fornece o conhecimento sobre como combinar seus elementos para produzir as informações em alto nível. Por exemplo, para originar o método de acesso em alto nível `getVolume`, para uma classe que represente a entidade `IfcWall`, a compilação associaria as definições do esquema ifcXML original com outro contendo associações entre os atributos necessários para calcular o resultado.

Idealmente, o próprio processo de geração do esquema complementar deveria ser automatizado, a partir de ontologias que definissem o que é uma parede, o que compõe uma parede, e como o conhecimento sobre a sua construção é agregado no modelo ifcXML. Porém, desenvolvimentos nesse sentido ainda são muito preliminares. Antes que as ontologias permitissem a automatização da geração do esquema complementar, essa operação poderia ser realizada em um ambiente gráfico. O desenvolvedor selecionaria entidades e atributos que pretendesse associar, e os processamentos a serem realizados pelas classes a serem geradas. Essa informação daria origem a metaclasses de conhecimento, e estas, por sua vez, seriam transcritas para o esquema complementar a ser compilado. Posteriormente, com o desenvolvimento de ontologias, essas operações seriam automatizadas. O esquema de criação dessas classes de acesso em alto nível seguiria esta sequência:

1. Em uma aplicação com interface gráfica, semelhante às utilizadas com EXPRESS-G, RDF ou OWL, o usuário selecionaria atributos de entidades do esquema ifcXML;
2. Estes atributos seriam associados por objetos representando o processamento necessário para o resultado esperado para a classe (por exemplo, área da seção multiplicada pela altura);
3. Diferentes atributos processamentos são encapsulados em metaclasses que representam o conhecimento sobre a manipulação de dados em alto nível;
4. As metaclasses são transcritas para um esquema XSD;
5. Em um metacompilador, como XMLBeans ou Java CC, o esquema XSD gerado pela aplicação e o esquema ifcXML são compilados, dando origem a bibliotecas de classes Java;
6. As bibliotecas são importadas pelas aplicações que acessarão os modelos de edifícios e processarão os seus dados.

Desse modo, garante-se a flexibilidade dos esquemas, e ao mesmo tempo automatiza-se consideravelmente a geração de classes. A biblioteca de classes pode obtida ser utilizada em várias aplicações, reduzindo o trabalho de programação. De fato, não é necessário que todos os programadores realizem estas etapas. Pode-se imaginar vários níveis de acesso à metacompilação, e o desenvolvimento de

bibliotecas modulares, que atendam as diferentes necessidades dos desenvolvedores de aplicações para as diferentes disciplinas da construção.

Por exemplo, pode ser desenvolvida uma biblioteca genérica para acesso a geometria em alto nível, que pode ser utilizada por aplicações orientadas para qualquer disciplina da indústria. Ou então bibliotecas especializadas, para atuar em conjunto com as primeiras e fornecer funcionalidades próprias de cada disciplina. Pode-se imaginar uma biblioteca de classes que atuassem como pré-processador para aplicações de desempenho físico, por exemplo.

Desenvolvedores poderiam acessar essas bibliotecas em nível intermediário, fazendo ajuste nas associações ou refinando os processamentos, dando origem a uma biblioteca mais adequada para as suas necessidades. Finalmente, um número maior de programadores sequer precisaria realizar a tarefa de metacompilação, pois um grande número de bibliotecas de classes de acesso em alto nível poderia ser disponibilizado, e a sua combinação seria suficiente para a maioria dos desenvolvimentos de aplicativos de acesso ao modelo.

Os trabalhos sobre metamodelagem e compilação de esquemas IFC e sobre reconstrução geométrica a partir modelos apresentados nesta dissertação são fontes sugeridas para o início dos estudos.

6
Referências

AMBROZEWICZ, P. H. L. **Qualidade na Prática: Conceitos e Ferramentas**. Curitiba: Serviço Nacional de Aprendizagem Industrial - Departamento Regional do Paraná, 2003.

APACHE. **XMLBeans** (página da internet).2008. http://xmlbeans.apache.org/, acessado em 12.2008.

ARANDA-MENA, G. e WAKEFIELD, R. **Interoperability of building information: myth or reality?** In: European Conference of Product and Process Modelling, 2006, Valencia. eWork and eBusiness in Architecture, Engineering and Construction. London: Taylor & Francis, 2006. Disponível em http://mams.rmit.edu.au/lrss7jid7nd4.pdf, acessado em 12.2008.

ARCHICLUBE. **Clube Brasileiro de Usuários de ArchiCAD** (página da internet).2008. www.archiclube.com.br, acessado em 12.2008.

AUGENBROE, G. **Trends in building simulation**. Building and Environment, v. 37, n. 8-9, 2002, p. 891-902. Disponível em http://dx.doi.org/10.1016/S0360-1323(02)00041-0, acessado em 12.2008.

AUTODESK. **Autodesk Developer Center** (página da internet).2008. http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=472012, acessado em 12.2008.

AYRES, C.*, et al.* **Utilização de CAD BIM para projeto de alvenaria de blocos de concreto**. In: Workshop Brasileiro de Gestão do Processo de Projetos na Construção de Edifícios, 8, 2008a, São Paulo. Anais. Disponível em http://www.arquitetura.eesc.usp.br/workshop08/secundarias/ANAIS/Artigo_08.pdf, acessado em 01.2009.

AYRES, C. e SCHEER, S. **Diferentes abordagens do uso do CAD no processo de projeto arquitetônico**. In: Workshop Brasileiro de Gestão do Processo de Projetos na Construção de Edifícios, 7, 2007, Curitiba. Disponível em http://www.cesec.ufpr.br/workshop2007/Artigo-57.pdf, acessado em 12.2008.

AYRES, C.*, et al.* **CAD-BIM requirements for masonry design process of concrete blocks.** In: CIB W78 International Conference on Information Technology on Information Technology, 25, 2008b, Santiago. Proceedings. Universidad de Talca e Stanford University p. 40-47. acessado em 12.2008.

BANDURSKI, A. E. e JEFFERSON, D. K. **Data description for computer-aided design**. In: International Conference on Management of Data, 3, 1975, San Jose. Proceedings of the 1975 ACM SIGMOD international conference on Management of data. New York: ACM, 1975. p. 193 - 202. Disponível em http://portal.acm.org/citation.cfm?id=500080.500107&coll=portal&dl=ACM&CFID=15982252&CFTOKEN=17235559, acessado em 12.2008.

BEALL, C. **Masonry Design and Detailing**. 5 ed. New York: McGraw-Hill, 2003, 640 p.

BÉDARD, C. **On the adoption of computing and IT by industry: the case for integration in early building design**. In: Intelligent Computing in Engineering and Architecture, 13, 2006, Ascona. Lecture

Notes in Computer Science. Berlin: Springer, 2006. p. 62-73. Disponível em http://www.springerlink.com/content/m524h81343w8760l/, acessado em 12.2008.

BERARD, O. e SHULZ, J. **Geometrical correction and conversion of IFC2X3 models**. Copenhagen, 2008, 127 p. Dissertação (Mestrado) - IT University of Copenhagen. Disponível em http://www.bitu.dk/download/Masters%20Thesis%20-%20Afleveret.pdf, acessado em 12.2008.

BERTEZINI, A. L. **Métodos de avaliação do processo de projeto de arquitetura na construção de edifícios sob a ótica da gestão da qualidade**. São Paulo, 2006, 208 p. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo. Disponível em http://www.infohab.org.br, acessado em 20.11.2006.

BEUCKE, K.*, et al.* **Applications of virtual design and construction in the building industry**. Structural Engineering International, v. 15, n. 3, 2005, p. 129-134. Disponível em http://dx.doi.org/10.2749/101686605777962973, acessado em 12.2008.

BIRX, G. W. **How Building Information Modelling changes architectural practice**. The American Institute of Architects - Best Practices, 2006. Disponível em http://www.aia.org/bestpractices_index, acessado em 22.11.2006.

BJÖRK, B.-C. **A conceptual model of spaces, space boundaries and enclosing structures**. Automation in Construction, v. 1, n. 3, 1992, p. 193-214. Disponível em http://dx.doi.org/10.1016/0926-5805(92)90013-A, acessado em 12.2008.

BRUNNERMEIER, S. B. e MARTIN, S. A. **Interoperability Cost Analysis of the US Automotive Supply Chain - Final Report**. Research Triangle Institute, 1999, 93 p. Disponível em http://www.rti.org/pubs/US_Automotive.pdf. Acessado em: 11.2008.

CAMBIAGHI, H. **Diretrizes Gerais para a Intercambialidade de projetos com CAD**. São Paulo: Pini, 2000, 44 p.

CHASTAIN, T.*, et al.* **Square peg in a round hole or horseless carriage? Reflections on the use of computing in architecture**. Automation in Construction, v. 11, n. 2, 2002, p. 237-248. Disponível em http://dx.doi.org/10.1016/S0926-5805(00)00095-9, acessado em 12.2008.

COONS, S. A. **An outline of the requirements for a computer-aided design system**. In: Spring Joint Computer Conference, 1963, Detroit. Proceedings. New York: ACM, 1963. p. 299-304. Disponível em http://doi.acm.org/10.1145/1461551.1461588, acessado em 11.2008.

CSIRO. **LCADesign** (página da internet).2008. http://www.cmmt.csiro.au/brochures/tech/lcadesign/, acessado em 01.2009.

DÉCIO, O. C. **XML**. São Paulo: Novatec, 2000, 96 p.

EASTMAN, C. M. **General purpose building description systems**. Computer-Aided Design, v. 8, n. 1, 1976, p. 17-26. Disponível em http://dx.doi.org/10.1016/0010-4485(76)90005-1, acessado em 11.2008.

EASTMAN, C. M. **Information and databases in design**. Design Studies, v. 1, n. 3, 1980a, p. 146-152. Disponível em http://dx.doi.org/10.1016/0142-694X(80)90021-6, acessado em 11.2008.

EASTMAN, C. M. **Prototype integrated building model**. Computer-Aided Design, v. 12, n. 3, 1980b, p. 115-119. Disponível em http://dx.doi.org/10.1016/0010-4485(80)90003-2, acessado em 11.2008.

EASTMAN, C. M. **Architectural CAD: a ten year assessment of the state of the art**. Computer-Aided Design, v. 21, n. 5, 1989, p. 289-292. Disponível em http://dx.doi.org/10.1016/0010-4485(89)90034-1, acessado em 11.2008.

EASTMAN, C. M. **The evolution of CAD: integrating multiple representations**. Building and Environment, v. 26, n. 1, 1991, p. 17-23. Disponível em http://dx.doi.org/10.1016/0360-1323(91)90035-A, acessado em 12.2008.

EASTMAN, C. M. **Modeling of buildings: evolution and concepts**. Automation in Construction, v. 1, n. 2, 1992, p. 99-109. Disponível em http://dx.doi.org/10.1016/0926-5805(92)90001-Z, acessado em 12.2008.

EASTMAN, C. M. **Building Product Models: Computer Environments Supporting Design and Construction**. Boca Raton: CRC Press, 1999, 424 p.

EASTMAN, C. M. **New methods of architecture and building**. ACADIA Conference 2004, Toronto. Disponível em http://bim.arch.gatech.edu/data/reference/New%20Methods%20of%20Architecture%20and%20Building_ACADIA2004.pdf, acessado em 12.2008.

EASTMAN, C. M. e HENRION, M. **Glide: a language for design information systems**. In: SIGGRAPH - International Conference on Computer Graphics and Interactive Techniques, 4, 1977, San Jose. Proceedings. New York: ACM, 1977. p. 23-33. Disponível em http://doi.acm.org/10.1145/563858.563863, acessado em 11.2008.

EASTMAN, C. M. e HENRION, M. **Geometric modelling - a survey**. Computer-Aided Design, v. 11, n. 5, 1979, p. 253-272. Disponível em http://dx.doi.org/10.1016/0010-4485(79)90071-X, acessado em 12.2008.

EASTMAN, C. M. e KUTAY, A. **Transaction management in design databases**. In: MIT-JSME Workshop, 1989, Cambridge. Computer-Aided Cooperative Product Development. Berlin: Springer, 1991. p. 334-351. Disponível em http://www.springerlink.com/content/75007584l378r29v/fulltext.pdf, acessado em 12.2008.

EASTMAN, C. M.*, et al.* **Functional modeling in parametric CAD systems**. In: ACADIA Conference 2004, 2004, Toronto. Disponível em http://bim.arch.gatech.edu/data/reference/Functional%20modeling%20in%20parametric%20CAD%20systems_GCAD2004.pdf, acessado em 12.2008.

EASTMAN, C. M.*, et al.* **BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors**. Hoboken: Wiley, 2008, 490 p.

FABRICIO, M. M.*, et al.* **Estudo da seqüência de etapas do projeto na construção de edifícios: cenários e perspectivas**. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO, 18, 1998, Niterói, RJ. Disponível em http://www.eesc.sc.usp.br/sap/docentes/fabricio/ENEGEP98-ES_Fluxo_Proj.pdf, acessado em 20.11.2006.

FABRICIO, M. M.*, et al.* **Brief Reflection on Improvement of Design Process Efficiency in Brazilian Building Projects**. In: International Group for Lean Construction Conference, 8, 1999, Berkeley.

Proceedings. p. 345-356. Disponível em http://www.iglc.net/conferences/1999/Papers/, acessado em 12.2008.

FAVRE, D. e CITHERLET, S. **Eco-Bat: A design tool for assessing environmental impacts of buildings and equipment** Building Simulation, v. 1, n. 1, 2008, p. 83-94. Disponível em http://www.springerlink.com/content/xg23588701l273ql/, acessado em 12.2008.

FENVES, S. J. **A Core Product Model for Representing Design Information - NISTIR 6736**. Boca Raton: NIST - National Institute of Standards and Technology, 2002, 424 p. Disponível em http://www.mel.nist.gov/msidlibrary/doc/ir6736.pdf. Acessado em: 12.2008.

FOWLER, J. **STEP for Data Management Exchange and Sharing**. Twickenham: Technology Appraisals, 1996, 222 p. Disponível em http://www.pdtsolutions.co.uk/book/STEPbook.pdf. Acessado em: 12.2008.

GALLAGER, R. G. e MITTER, S. K. **From Servo Loops to Fiber Nets**. LIDS 50th Anniversary Symposium, Massachusetts Institute of Technology - The Laboratory for Information and Decision Systems, Cambridge, 25.10.1990. Disponível em http://lidsblog.typepad.com/lidsblog/files/Servoloops_Fibernets.pdf, acessado em 11.2008.

GALLAHER, M. P.*, et al.* **Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry**. Research Triangle Institute, 2004, 210 p. Disponível em http://www.bfrl.nist.gov/oae/publications/gcrs/04867.pdf. Acessado em: 12.2008.

GIELINGH, W. **An assessment of the current state of product data technologies**. Computer-Aided Design, v. 40, n. 7, 2008, p. 750-759. Disponível em http://dx.doi.org/10.1016/j.cad.2008.06.003, acessado em 12.2008.

GINGERICH, J. Z. **Computer graphics building definition system**. In: Annual ACM IEEE Design Automation Conference, 10, 1973. Proceedings of the 10th Workshop on Design Automation. Piscataway: IEEE Press, 1973. Disponível em http://portal.acm.org/toc.cfm?id=800124&type=proceeding&coll=portal&dl=ACM&CFID=15394702&CFTOKEN=69261272, acessado em 12.2008.

GRABOWSKI, H. e ANDERL, R. **Integration of the design and manufacture planning process based on a CAD system with a technology oriented volume model**. Computers & Graphics, v. 7, n. 2, 1983, p. 125-141. Disponível em http://dx.doi.org/10.1016/0097-8493(83)90003-1, acessado em 12.2008.

GRAPHISOFT. **ArchiCAD 10 GDL Reference Guide**. Graphisoft, 2006, 320 p.

GRAPHISOFT. **ArchiCAD** (página da internet).2008a. http://www.graphisoft.com/products/archicad/, acessado em 12.2008.

GRAPHISOFT. **ArchiCAD 12 GDL Reference Guide**. Graphisoft, 2008b, 336 p.

GRAPHISOFT. **Graphisoft API Development Kit Documentation** (página da internet).2008c. http://www.graphisoft.com/support/developer/documentation/DocAPIDevKit.html, acessado em 01.2009.

GRAPHISOFT. **Graphisoft Application Programming Interface Developer Program** (página da internet).2008d. http://www.graphisoft.com/support/developer/api.html, acessado em 01.2009.

GRAPHISOFT. **Graphisoft Developer Center** (página da internet).2008e. http://www.graphisoft.com/support/developer/, acessado em 12.2008.

GRAPHISOFT. **Graphisoft GDL Technical Standards v 2.0** (página da internet).2008f. http://www.graphisoft.com/ftp/techsupport/documentation/developer_docs/BasicLibraryDoc/10/LibDevGuide/TechnicalStandards.html, acessado em 12.2008.

HANNUS, M. **Implementation of object oriented product model applications**. In: CIB W78 Workshop: The Computer Integrated Future, 1991, Eidenhoven. Disponível em http://itc.scix.net/cgi-bin/works/Show?w78-1991-10, acessado em 12.2008.

HARTMANN, T. e FISCHER, M. **Applications of BIM and hurdles for widespread adoption of BIM - 2007 AISC-ACCL eConstruction Roundtable Event Report**. Stanford: CIFE/Stanford University, 2008, 20 p. Disponível em http://cife.stanford.edu/online.publications/WP105.pdf. Acessado em: 12.2008.

HAYNIE, M. N. **Tutorial: the relational data model for design automation**. In: Annual ACM IEEE Design Automation Conference, 20, 1983, Miami Beach. Proceedings. New York: ACM, 1983. p. 599-607. Disponível em http://portal.acm.org/citation.cfm?id=800032.800731&coll=GUIDE&dl=GUIDE&CFID=14803425&CFTOKEN=31163327, acessado em 12.2008.

HIETANEN, J. e DROGEMULLER, R. **Approaches to university level BIM education**. In: IABSE Conference, 2008, Helsinki. Proceedings.

HOFFMANN, C. M. e JOAN-ARINYO, R. **Parametric modeling**. In: FARIN, G.*, et al*. Handbook of Computer Aided Geometric Design. Amsterdam: Elsevier, 2002, p. 519-541. Disponível em http://dx.doi.org/10.1016/B978-044451104-1/50022-8. Acessado em: 11.2008.

HOSAKA, M. e KIMURA, F. **A model-based approach to CAD/CAM integration**. Computers in Industry, v. 14, n. 1, 1990, p. 35-42. Disponível em http://dx.doi.org/10.1016/0166-3615(90)90101-T, acessado em 12.2008.

HOWARD, R. e BJÖRK, B.-C. **Use of standards for CAD layers in building**. Automation in Construction, v. 16, n. 3, 2007, p. 290-297 Disponível em http://dx.doi.org/10.1016/j.autcon.2006.06.001, acessado em 12.2008.

HOWARD, R. e BJÖRK, B.-C. **Building information modelling - Experts' views on standardisation and industry deployment**. Advanced Engineering Informatics, v. 22, n. 2, 2008, p. 271-280. Disponível em http://dx.doi.org/10.1016/j.aei.2007.03.001, acessado em 11.2008.

HVAM, L. **A procedure for the application of product modelling**. International Journal of Production Research, v. 39, n. 5, 2001, p. 873-885. Disponível em http://www.ingentaconnect.com/content/tandf/tprs/2001/00000039/00000005/art00004, acessado em 12.2008.

IAI. **IFC Tools for Developers** (página da internet).2008a. http://www.iai-international.org/software/Tools%20for%20IFC%20developers.html, acessado em 12.2008.

IAI. **IFD Library White Paper**. IAI, 2008b, 9 p. Disponível em http://www.ifd-library.org/images/IFD_Library_White_Paper_2008-04-10_I_.pdf. Acessado em: 12.2008.

IAI. **International Alliance for Interoperability** (página da internet).2008c. http://www.iai-international.org/, acessado em 12.2008.

IBRAHIM, M.*, et al.* **CAD smart objects: potentials and limitations**. In: International eCAADe Conference, 21, 2003, Graz. Proceedings. p. 547-551. Disponível em http://www.iit.edu/~krawczyk/miecad03.pdf, acessado em 12.2008.

IBRAHIM, M.*, et al.* **Two Approaches to BIM: A Comparative Study**. In: eCAADe Conference, 22, 2004, Copenhagen. Proceedings. p. 610-616. Disponível em http://cumincad.scix.net/cgi-bin/works/Show?2004_610, acessado em 12.2008.

ISIKDAG, U.*, et al.* **Building information models: a review on storage and exchange mechanisms**. In: CIB W78 International Conference on Information Technology on Information Technology, 24, 2007, Maribor. Proceedings. p. 135-144. Disponível em http://itc.scix.net/data/works/att/w78-2007-020-068b-Isikdag.pdf, acessado em 12.2008.

JANDL, P. **Java 6**. São Paulo: Novatec, 2008, 144 p.

JOHNSON, T. E. **Sketchpad III - a computer program for drawing in three dimensions**. In: Spring Joint Computer Conference, 1963, Detroit. Proceedings. New York: ACM, 1963. p. 347-353. Disponível em http://doi.acm.org/10.1145/1461551.1461592, acessado em 11.2008.

KALAY, Y. E. **A database management approach to CAD/CAM systems integration**. In: Annual ACM IEEE Design Automation Conference, 22, 1985a, Las Vegas. Proceedings. New York: ACM, 1985. p. 111-116. Disponível em http://portal.acm.org/citation.cfm?id=317843, acessado em 12.2008.

KALAY, Y. E. **Redefining the role of computers in architecture: from drafting/modelling tools to knowledge-based design assistants**. Computer-Aided Design, v. 17, n. 7, 1985b, p. 319-328. Disponível em http://dx.doi.org/10.1016/0010-4485(85)90165-4, acessado em 12.2008.

KALAY, Y. E. **The impact of information technology on design methods, products and practices**. Design Studies, v. 27, n. 3, 2005, p. 357-380. Disponível em http://dx.doi.org/10.1016/j.destud.2005.11.001, acessado em 12.2008.

KIM, I. e SEO, J. **Development of IFC modeling extension for supporting drawing information exchange in the model-based construction environment**. Journal of Computing in Civil Engineering, v. 22, n. 3, 2008, p. 159-169. Disponível em http://dx.doi.org/10.1061/(ASCE)0887-3801(2008)22:3(159), acessado em 12.2008.

KIMURA, F.*, et al.* **A study on product modelling for integration of CAD/CAM**. Computers in Industry, v. 5, n. 3, 1984, p. 239-252. Disponível em http://dx.doi.org/10.1016/0166-3615(84)90004-6, acessado em 12.2008.

KIVINIEMI, A. **Ten years of IFC development - why are we not yet there?** Espoo: VTT, 2006, 43 p. Disponível em http://cic.vtt.fi/projects/vbe-net/data/20060615_Ten_Years_of_IFC_Development_@_CIB-W78_Montreal_Keynote.pdf. Acessado em: 12.2008.

KIVINIEMI, A.*, et al.* **Review of the development and implementation of IFC compatible BIM**. Erabuild, 2008, 128 p. Disponível em http://www.deaca.dk/file/9497/Review%20of%20the%20Development%20and%20Implementation%20of%20IFC%20compatible%20BIM.pdf. Acessado em: 12.2008.

KOWALTOWSKI, D. C. C. K.*, et al.* **Reflexão sobre metodologias de projeto arquitetônico**. Ambiente Construído, v. 6, n. 2, 2006, p. 07-19. Disponível em http://www.antac.org.br/ambienteconstruido/, acessado em 20.11.2006.

KRAUSE, F. L.*, et al.* **Product modelling**. CIRP Annals - Manufacturing Technology, v. 42, n. 2, 1993, p. 695-706. Disponível em http://dx.doi.org/10.1016/S0007-8506(07)62532-3, acessado em 12.2008.

KVAN, T. **Collaborative design: what is it?** Automation in Construction, v. 9, n. 4, 2000, p. 409-415 Disponível em http://dx.doi.org/10.1016/S0926-5805(99)00025-4, acessado em 12.2008.

LACROIX, M. e PIROTTE, A. **Data structures for CAD object description**. In: Annual ACM IEEE Design Automation Conference, 18, 1981, Nashville. Proceedings. New York: ACM, 1981. p. 653-659. Disponível em http://portal.acm.org/citation.cfm?id=802371, acessado em 12.2008.

LAM, P. T. I.*, et al.* **Contributions of designers to improving buildability and constructability**. Design Studies, n. 27, 2005, p. 457-479. Disponível em http://www.elsevier.com/locate/destud, acessado em 20.11.2006.

LEE, G.*, et al.* **Specifying parametric building object behavior (BOB) for a building information modeling system**. Automation in Construction, v. 15, n. 6, 2006, p. 758-776 Disponível em http://dx.doi.org/10.1016/j.autcon.2005.09.009, acessado em 12.2008.

LIEBICH, T.*, et al.* **Industry Foundation Classes IFC2x Edition 3**. IAI, 2006 Disponível em http://www.iai-international.org/Model/R2x3_final/index.htm. Acessado em: 12.2008.

LIEBICH, T. e WIX, J. **IFC Technical Guide**. IAI, 2000, 46 p. Disponível em http://www.iai-international.org/. Acessado em: 12.2008.

LKSOFTWARE. **Express Compiler** (página da internet).2008a. http://www.jsdai.net/eclipse/express-compiler, acessado em 12.2008.

LKSOFTWARE. **JSDAI Express API** (página da internet).2008b. http://www.jsdai.net/, acessado em 12.2008.

LOVE, P. E. D.*, et al.* **Researshing the investment of information technology in construction: an examination of evaluation practices**. Automation in Construction, n. 14, 2005, p. 569-582. Disponível em http://www.elsevier.com/locate/autcon, acessado em 09.12.2006.

MAHDAVI, A. **Computational building models: theme and four variations**. In: International IBPSA Conference, 8, 2003, Eindhoven. Proceedings. p. 3-17. Disponível em http://www.ibpsa.org/proceedings/BS2003/BS03_0003_18.pdf, acessado em 12.2008.

MARCOS, M.*, et al.* **Avaliação e análise de acessibilidade de um deficiente físico motor, através do software CATIA, em habitações de interesse social**. In: Workshop Brasileiro de Gestão do Processo de Projetos na Construção de Edifícios, 7, 2007, Curitiba. e-anais. Disponível em http://www.cesec.ufpr.br/workshop2007/Artigo-25.pdf, acessado em 12.2008.

MARTINS, O. S. **Determinação do potencial de seqüestro de carbono na recuperação de matas ciliares na região de São Carlos, SP**. São Carlos, 2004, 161 p. Tese (Doutorado) - Centro de Ciências Biológicas e da Saúde, Universidade Federal de São Carlos. Disponível em http://www.bdtd.ufscar.br/tde_busca/arquivo.php?codArquivo=816, acessado em 01.2009.

MELHADO, S. B. **Qualidade do projeto na construção de edifícios: aplicação ao caso das empresas de incorporação e construção**. São Paulo, 1994, 310 p. Tese (Doutorado) - Escola Politécnica Universidade de São Paulo.20.11.2006.

MELHADO, S. B. e AQUINO, J. P. R. D. **Perspectivas da utilização generalizada de Projetos para Produção na construção de edifícios**. In: GESTÃO DO PROCESSO DE PROJETO NA CONSTRUÇÃO DE EDIFÍCIOS, 2001, São Carlos, SP. Disponível em http://www.eesc.sc.usp.br/sap/workshop/anais/PERSPECTIVAS_DA_UTILIZACAO_GENERALIZADA_DE_PROJ_PARA_PRODUCAO.pdf, acessado em 20.11.2006.

MELHADO, S. B. e GRILO, L. M. **Desafios e oportunidades para os escritórios de projeto frente às tendências para a gestão do processo de projeto e do empreendimento**. Boletim Técnico da Escola Politécnica da USP - Departamento de Engenharia de Construção Civil, n. 336, 2003. Disponível em http://alunospos.pcc.usp.br/leonardo.grilo/Boletim_t%25C3%25A9cnico_com_mudan%25C3%25A7as.pdf, acessado em 20.11.2006.

MIKALDO, J. **Estudo comparativo do processo de compatibilização de projetos em 2D e 3D com uso de TI**. Curitiba, 2006, 150 p. Dissertação (Mestre) - Programa de Pós-Graduação em Construção Civil, Universidade Federal do Paraná. Disponível em http://www.ppgcc.ufpr.br/dissertacoes/d0073.PDF, acessado em 12.2008.

MOUM, A. **A framework for exploring the ICT impact on the architectural design process**. Electronic Journal of Information Technology in Construction, v. 11, 2006, p. 409-425. Disponível em http://www.itcon.org/data/works/att/2006_30.content.07890.pdf, acessado em 12.2008.

NASCIMENTO, L. A. e SANTOS, E. T. **Barreiras para o uso da tecnologia da informação na indústria da construção civil**. In: WORKSHOP NACIONAL GESTÃO DO PROCESSO DE PROJETO NA CONSTRUÇÃO DE EDIFÍCIOS, 2, 2002, Porto Alegre, RS. Disponível em http://www.infohab.org.br, acessado em 20.11.2006.

NASCIMENTO, L. A. D. e SANTOS, E. T. **A indústria da construção na era da informação**. Ambiente Construído, v. 3, n. 1, 2006, p. 69-81. Disponível em http://www.antac.org.br, acessado em 21.11.2006.

NEMETSCHEK. **Allplan** (página da internet).2008. http://www.allplan.com/, acessado em 12.2008.

NIBS. **National Building Information Modeling Standard**. National Institute of Building Sciences, 2007, 183 p. Disponível em http://nbims.opengeospatial.org/files/?artifact_id=742. Acessado em: 12.2008.

NICHOLSON-COLE, D. **Introduction to Object Making**. 2 ed. Graphisoft, 2004, 124 p. Disponível em http://tr.graphisoftus.com/downloads/DL%27s%20for%20GraphisoftUS/Introduction%20to%20Object%20Making/Intro%20to%20Object%20Making.pdf. Acessado em: 12.2008.

NILSSON, P. e FAGERSTRÖM, B. **Managing stakeholder requirements in a product modelling system**. Computers in Industry, v. 57, n. 2, 2006, p. 167-177. Disponível em http://dx.doi.org/10.1016/j.compind.2005.06.003, acessado em 12.2008.

NISBET, N. e LIEBICH, T. **ifcXML Implementation Guide**. 2 ed. International Alliance for Interoperability - Modeling Support Group, 2007, 50 p. Disponível em http://www.iai-tech.org/products/ifc_specification/ifcxml-releases. Acessado em: 12.2008.

NOUR, M. M. **A Flexible Model for Incorporating Construction Product Data into Building Information Models**. Weimar, 2006, 186 p. Tese (Doutorado) - Construction Informatics, Bauhaus. Disponível em http://e-pub.uni-weimar.de/volltexte/2006/778/, acessado em 12.2008.

NOUR, M. M. e BEUCKE, K. **An Open Platform for Processing IFC Model Versions**. Tsinghua Science and Technology, v. 13, n. S1, 2008, p. 126-131. Disponível em http://qhxb.lib.tsinghua.edu.cn/myweb/english/2008/2008es1/126-131.pdf, acessado em 12.2008.

NOVAES, C. C. **Diretrizes para garantia da qualidade do projeto na produção de edifícios habitacionais**. São Paulo, 1996, 280 p. Tese (Doutorado) - Escola Politécnica, Universidade de São Paulo. Disponível em http://www.infohab.org.br, acessado em 20.11.2006.

PAULA, A. T. D. e MELHADO, S. B. **Avaliação do impacto potencial da versão 2000 das normas ISO 9000 na gestão e certificação da qualidade: o caso das empresas construtoras**. Boletim Técnico da Escola Politécnica da USP - Departamento de Engenharia de Construção Civil, n. 395, 2005. Disponível em http://www.infohab.org.br, acessado em 03.12.2006.

PAZLAR, T. e TURK, Z. **Interoperability in practice: geometric data exchange using the IFC standard**. International Journal of Production Research, v. 13, 2008, p. 362-380. Disponível em http://www.itcon.org/data/works/att/2008_24.content.00881.pdf, acessado em 12.2008.

PEANSUPAP, V. e WALKER, D. H. T. **Factors enabling information and communication technology diffusion and actual implementation in construction organisations**. Electronic Journal of Information Technology in Construction, v. 10, n. 14, 2005, p. 193-218. Disponível em http://www.itcon.org/2005/14/, acessado em 09.12.2006.

PELS, H. J. **Product and process data modelling**. Computers in Industry, v. 31, n. 3, 1996, p. 191-194. Disponível em http://dx.doi.org/10.1016/S0166-3615(96)00050-4, acessado em 12.2008.

PENTTILA, H. **The state of the art of Finnish building product modelling methodology**. In: Computer Aided Architectural Design Futures Conference, 11, 2005, Viena. Learning from the Past a Foundation for the Future. Viena: p. 225-240. Disponível em http://cumincad.scix.net/data/works/att/cf2005_2_55_233.content.pdf, acessado em 12.2008.

PFEIFER, G.*, et al.* **Masonry Construction Manual**. Berlin: Birkhauser, 2001, 392 p.

PINI. **TCPO - Tabelas de Composição de Preços para Orçamentos**. 12 ed. São Paulo: Pini, 2003, 441 p.

PLESSIS, C. D. **Agenda 21 for Sustainable Construction in Developing Countries**. Pretoria: CSIR, 2001, 55 p. Disponível em http://www.worldarchitecture.org/links/?http://www.sustainablesettlement.co.za/docs/a21_discussiondoc.pdf.

PRÉ. **SimaPro LCA** (página da internet).2008. http://www.pre.nl/simapro/simapro_lca_software.htm, acessado em 12.2008.

PRUDÊNCIO, L. R.*, et al.* **Alvenaria Estrutural de Blocos de Concreto**. Florianópolis: Palotti, 2002, 208 p.

PSQ. **Programa Setorial da Qualidade: Setor de Projetos**. São Paulo: ASBEA/ABECE/ABRASIP/IAB-SP/IE/SINAECO/SINDINSTALAÇÃO, 1997.

RAMALHO, M. A. e CORRÊA, M. R. S. **Projeto de Edifícios de Alvenaria Estrutural**. São Paulo: Pini, 2003, 174 p.

RIVARD, H. **A survey on the impact of information technology on the canadian architecture, engineering and construction industry**. Electronic Journal of Information Technology in

Construction, v. 5, n. 3, 2000, p. 37-56. Disponível em http://itcon.org/2000/3/, acessado em 09.12.2006.

ROAF, S. **Closing the Loop: Benchmarks for Sustainable Buildings**. London: Riba Publications, 2004, 532 p.

ROAF, S. e DAY, C. **Eco-House: A design guide**. Oxford: Architectural Press, 2001.

ROBINSON, C. E. **A data structure for a computer aided design system**. In: Annual ACM IEEE Design Automation Conference, 3, 1966, New York. Proceedings of the SHARE design automation project. New York: ACM, 1966. p. 14.1-14.9. Disponível em http://portal.acm.org/citation.cfm?id=810786, acessado em 12.2008.

ROMANO, F. V.*, et al.* **A importância da modelagem do processo de projeto para o desenvolvimento integrado de edificações**. In: GESTÃO DO PROCESSO DE PROJETO NA CONSTRUÇÃO DE EDIFÍCIOS, 2001, São Carlos, SC. Disponível em http://www.infohab.org.br, acessado em 20.11.2006.

ROSS, D. T. **Computer-aided design (research summary)**. Communications of the ACM, v. 4, n. 5, 1961, p. 235. Disponível em http://doi.acm.org/10.1145/366532.366554, acessado em 11.2008.

ROSS, D. T. e RODRIGUEZ, J. E. **Theoretical foundations of the computer-aided design system**. In: Spring Joint Computer Conference, 1963, Detroit. Proceedings. New York: ACM, 1963. p. 305-322. Disponível em http://doi.acm.org/10.1145/1461551.1461589, acessado em 12.2008.

SACKS, R. e BARAK, R. **Impact of three-dimensional parametric modeling of buildings on productivity in structural engineering practice**. Automation in Construction, v. 17, n. 4, 2007, p. 439-449 Disponível em http://dx.doi.org/10.1016/j.autcon.2007.08.003, acessado em 12.2008.

SACKS, R.*, et al.* **Parametric 3D modeling in building construction with examples from precast concrete** Automation in Construction, v. 13, n. 3, 2004, p. 291-312. Disponível em http://dx.doi.org/10.1016/S0926-5805(03)00043-8, acessado em 12.2008.

SAXPROJECT. **SAX - Simple API for XML** (página da internet).2008. http://www.saxproject.org/, acessado em 12.2008.

SCHEER, S.*, et al.* **Impactos do uso do sistema CAD geométrico e do uso do sistema CAD BIM no processo de projeto em escritórios de arquitetura**. In: Workshop Brasileiro de Gestão do Processo de Projetos na Construção de Edifícios, 7, 2007, Curitiba. Disponível em http://www.cesec.ufpr.br/workshop2007

SCHILDT, H. **Java 2 - The Complete Reference**. 5 ed. New York: McGraw-Hill, 2002, 1156 p.

SCHMID, A. L. **Simulação de desempenho térmico em múltiplas zonas: mestre, um sistema brasileiro na linguagem Java**. In: Encontro Nacional de Conforto no Ambiente Construído, 6, 2001, São Pedro. Anais. Disponível em http://www.infohab.org.br, acessado em 01.2009.

SCHMID, A. L. **Simulação da luz natural: combinação dos algoritmos de raytracing e radiosidade e aplicações na arquitetura**. Ambiente Construído, v. 4, n. 3, 2004, p. 51-59. Disponível em http://www.antac.org.br/ambienteconstruido/pdf/revista/artigos/Doc117117.pdf, acessado em 01.2009.

SCHMID, A. L. **Acústica arquitetônica e auralização no sistema Mestre de simulação de edifícios**. In: Encontro Anual da Sociedade Brasileira de Acústica, 27, 2006, São Paulo. Anais.

SCHMID, A. L. e AYRES, C. **Testes iniciais do sistema de modelagem ArchiCAD como pré-processador para o sistema Mestre de análise térmica, lumínica e acústica de edificações**. In: Encontro Nacional de Conforto no Ambiente Construído, 9, 2007, Ouro Preto. Anais. p. 1697-1702.

SCLCI. **Swiss Centre for Life Cycle Inventories - The Ecoinvent Database** (página da internet).2008. http://www.ecoinvent.org/, acessado em 12.2008.

SCRA-STEP. **STEP Application Handbook - ISO 10303 - Version 3**. North Charleston: SCRA, 2006, 175 p.

SOUZA, U. E. L. D. **Como reduzir perdas nos canteiros**. São Paulo: Pini, 2005, 128 p.

SPERLING, D. M. **O projeto arquitetônico, novas tecnologias de informação e o museu Guggenhein de Bilbao**. In: Workshop Nacional Gestão do Processo de Projeto na Construção de Edifícios, 2, 2002, Porto Alegre. e-anais. Disponível em http://www.eesc.sc.usp.br/sap/projetar/files/A038.pdf, acessado em 12.2008.

STOUFFS, R. **Constructing design representations using a sortal approach**. Advanced Engineering Informatics, v. 22, n. 1, 2008, p. 71-89. Disponível em http://dx.doi.org/10.1016/j.aei.2007.08.007, acessado em 12.2008.

SUTHERLAND, I. E. **Sketchpad - a man-machine graphical communication system**. In: Spring Joint Computer Conference, 1963, Detroit. Proceedings. New York: ACM, 1963. p. 329-346. Disponível em http://doi.acm.org/10.1145/800265.810742, acessado em 12.2008.

TAKEDA, H.*, et al.* **Modeling design processes**. AI Magazine, v. 11, n. 4, 1990, p. 37-48. Disponível em http://www.aaai.org/ojs/index.php/aimagazine/article/view/855/773, acessado em 12.2008.

TAVARES, S. F. **Metodologia de Análise do Ciclo de Vida Energético de Edificações Residenciais Brasileiras**. Florianópolis, 2006, 225 p. Tese (Doutorado) - Programa de Pós-Graduação em Engenharia Civil, Universidade Federal de Santa Catarina. Disponível em http://tede.ufsc.br/teses/PECV0459.pdf, acessado em 12.2008.

TNO. **IFC Engine DLL API Reference Guide** (página da internet).2008a. http://www.ifcbrowser.com/downloads/BETA/IFCEngineDLL.API.doc, acessado em 12.2008.

TNO. **Reading IFC Example** (página da internet).2008b. http://www.ifcbrowser.com/ifcenginedllread.html, acessado em 12.2008.

TNO. **TNO IFC Engine Series** (página da internet).2008c. http://www.ifcbrowser.com/, acessado em 12.2008.

TREECK, C. V.*, et al.* **Simulation based on the product model standard IFC**. In: International IBPSA Conference, 2003, Eindhoven. Proceedings. p. 1293-1300. Disponível em http://www.inive.org/members_area/medias/pdf/Inive%5CIBPSA%5CUFSC75.pdf, acessado em 12.2008.

TSE, T.-C. K.*, et al.* **The utilisation of building information models in nD modelling: A study of data interfacing and adoption barriers**. Electronic Journal of Information Technology in Construction, v. 10, 2005, p. 85-110. Disponível em http://www.itcon.org/data/works/att/2005_8.content.05676.pdf, acessado em 12.2008.

TUCKER, S.*, et al.* **LCADesign: an integrated approach to automatic eco-efficiency assessment of commercial buildings**. In: CIB W78 International Conference on Information Technology on

Information Technology, 20, 2003, Waiheke Island. Disponível em http://itc.scix.net/data/works/att/w78-2003-403.content.pdf, acessado em 12.2008.

TURK, Z. **Phenomenological foundations of conceptual product modelling in architecture, engineering and construction**. Artificial Intelligence in Engineering, v. 15, n. 2, 2001, p. 83-92. Disponível em http://dx.doi.org/10.1016/S0954-1810(01)00008-5, acessado em 12.2008.

TURK, Z. **Construction informatics: definition and ontology**. Advanced Engineering Informatics, v. 20, n. 2, 2006a, p. 187-199 Disponível em http://dx.doi.org/10.1016/j.aei.2005.10.002, acessado em 12.2008.

TURK, Z. **Methodologies for construction informatics research**. In: Intelligent Computing in Engineering and Architecture, 13, 2006b, Ascona. Lecture Notes in Computer Science. Berlin: Springer, 2006. p. 663-669. Disponível em http://www.springerlink.com/content/vq75m7821366g515/, acessado em 12.2008.

TZORTZOPOULOS, P. **Contribuições para o desenvolvimento de um modelo de processo de projeto de edificações em empresas construtoras incorporadoras de pequeno porte**. Porto Alegre, 1999, 163 p. Dissertação (Mestrado) - Departamento de Pós-graduação em Engenharia Civil, Universidade Federal do Rio Grande do Sul. Disponível em http://www.infohab.org.br, acessado em

TZORTZOPOULOS, P. **The design and implementation of product development process models in construction companies**. Salford, 2004, 321 p. Tese (Doutorado) - School of Construction and Property Management, University of Salford. Disponível em http://tede.ibict.br/tde_busca/processaArquivo.php?codArquivo=230, acessado em 12.2008.

VELOSO, R. R. **Java e XML**. 2 ed. São Paulo: Novatec, 2007, 109 p.

VÉRTESI, L.*, et al.* **ArchiCAD 4.0 Reference Manual**. Graphisoft, 1991, 163 p.

VOELCKER, H. B.*, et al.* **The PADL-1.0/2 system for defining and displaying solid objects**. ACM SIGGRAPH Computer Graphics, v. 12, n. 3, 1978, p. 257-263. Disponível em http://doi.acm.org/10.1145/800248.807400, acessado em 12.2008.

VTT. **Finnish ICT Barometer**. Espoo: TEKES, 2007 Disponível em http://cic.vtt.fi/projects/vbe-net/data/2007_ICT_barometer.pdf. Acessado em: 12.2008.

W3C. **DOM - Document Object Model** (página da internet).2008a. http://www.w3.org/DOM/, acessado em 12.2008.

W3C. **Extensible Markup Language - XML - 1.0, Fifth Edition** (página da internet).2008b. http://www.w3.org/TR/REC-xml/, acessado em 12.2008.

WAN, L. e MESSNER, J. I. **Virtual construction simulator: a 4D CAD model generation prototype**. In: Computing in Civil Engineering, 2007, Pittsburgh. Proceeding of the 2007 ASCE International Workshop on Computing in Civil Engineering Pennsylvania, 2007. Disponível em http://dx.doi.org/10.1061/40937(261)102, acessado em 12.2008.

WANG, W.-C.*, et al.* **Modeling of design iterations throug simulation**. Automation in Construction, n. 15, 2006, p. 589-603. Disponível em http://www.elsevier.com/locate/autcon, acessado em 09.12.2006.

WISSENBACH, V. **Manual Técnico de Alvenaria**. São Paulo: Projeto, 1990, 275 p.

WORTMANN, J. C. **Production management systems for one-of-a-kind products**. Computers in Industry, v. 19, n. 1, 1992, p. 79-88. Disponível em http://dx.doi.org/10.1016/0166-3615(92)90008-B, acessado em 12.2008.

YANG, W. Z.*, et al.* **Recent development on product modelling: a review**. International Journal of Production Research, v. 46, n. 1, 2008, p. 6055-6085. Disponível em http://www.ingentaconnect.com/content/tandf/tprs/2008/00000046/00000021/art00008, acessado em 12.2008.