SIGraDi2020 Transformative design
XXIV International Conference of the Iberoamerican Society of Digital Graphics
Medellín | Colombia

Integration of BIM and Algorithmic Design logics through data exchange between Grasshopper plugin and Revit and Archicad software

Marina Pires Iasbik
Universidade Federal de Viçosa | Brasil | marinaiasbik@gmail.com

Andressa Carmo Pena Martinez
Universidade Federal de Viçosa | Brasil | andressamartinez@gmail.com

Jorge Lira de Toledo Gazel
Universidade de São Paulo | Brasil | jorge.lira@usp.br

Abstract

The Algorithmic Design's integration with BIM (Building Information Modeling), allows greater potential for formal design innovation, tasks automation, greater geometry control, data assignment, and project documentation throughout its life cycle. This paper aims to assist in this integration, analyzing some plugins for conversion from Grasshopper to Archicad and Revit. Based on a parameterized social housing model, interoperability tests were carried out to compare different workflows and discuss some strategies and logics of algorithmic modeling to facilitate the communication between Grasshopper and BIM.

Keywords: Algorithm design; Building information modeling; Parametric modeling; Project process; Interoperability.

INTRODUÇÃO

Com os rebatimentos de processos digitais de projeto na arquitetura, arquitetos tem buscado conhecimentos cada vez mais diversos para atender às variadas necessidades, adaptações e transformações no campo da arquitetura (Feist, 2016). Segundo Larrondo (2017), o *design* algorítmico vem se tornando cada vez mais popular nesse cenário, apoiando a criação de formas variadas com relacionamentos parametricamente restritos. Schumacher (2016, p.08) exemplifica as mudanças positivas na forma de projetar trazidas pelos auxílios computacionais que constituem o *design* paramétrico e algorítmico:

> O princípio de *design* de "gerar e testar" realizado em um meio ou modelo de design antes da construção física é um substituto econômico (racional) para a "tentativa e erro", processo que é o princípio da evolução biológica, bem como de toda a evolução cultural pré-arquitetural. Tanto os poderes quanto os fatores da racionalidade do projeto estão sendo massivamente aprimorados pelos auxílios computacionais que constituem o design paramétrico e algorítmico em comparação com o design tradicional baseado no desenho de acordo com o precedente ou a intuição. (Schumacher, 2016, p.08, tradução nossa)

Juntamente ao design algorítmico, o uso do BIM (*Building Information Modeling*) tem se destacado em todo o ciclo de vida de um projeto, permitindo que a documentação bidimensional e a exploração formal tridimensional desenvolvam-se simultaneamente, tornando mais dinâmico esse processo e encurtando o caminho entre o modelo virtual e a representação gráfica (Castelo Branco e Leitão, 2017).

Todavia, os *software* mais utilizados para modelagem algorítmica em arquitetura não possuem direcionamento para gerar a representação gráfica e documentação técnica necessárias para a execução destes projetos (Guidoux Gonzaga et al., 2018). Portanto, faz-se relevante integrar estas duas atividades projetuais primordiais: a livre criação de formas possibilitada pelo *design* algorítmico e a documentação técnica necessária para execução do projeto arquitetônico, propiciada pelo BIM.

Nesse contexto, esse artigo visa auxiliar na integração das lógicas projetuais algorítmicas e de modelagem da informação para construção, tendo como objetivo principal testar as limitações da transmissão de dados entre um *software* de modelagem algorítmica e um *software* BIM. Para tanto, será feito um levantamento dos principais fluxos de importação de geometrias aos *software Archicad* e *Revit*, através de *plugins* de interoperabilidade, visando gerar um levantamento de estratégias de conversão entre *software* de modelagem algorítmica e BIM.

Esse trabalho constitui a quarta etapa de uma pesquisa sobre customização em massa para habitação de interesse social (Gazel et al., 2018), que consiste no processo digital das unidades, seguindo as seguintes etapas: (1) pesquisa preliminar de dados de perfil de usuários e modos de morar; (2) a modelagem conceitual arquitetônica; (3) a modelagem algorítmico-paramétrica das unidades no *Grasshopper*; (4) o desenvolvimento do modelo BIM; (5) definição de plataforma/ estratégias para escolhas personalizadas pelos usuários.

Essa quarta etapa consiste no estudo e aplicação dos conceitos de interoperabilidade entre o processo de projeto algorítmico-paramétrico e o BIM. Adicionalmente, também visa explorar a interação dos usuários com os parâmetros modificáveis nos *scripts*, cujas alterações são simultaneamente visualizadas na interface do *software*

BIM, tornando o processo também um sistema de suporte às decisões.

Nesse contexto, o conceito de customização em massa de moradias com a plataforma BIM, permite combinar objetivos controversos de individualização e produção econômica, permitindo a máxima flexibilidade no desenvolvimento de projetos com uma estrutura de custos reduzida. Dada a grande quantidade de conteúdo de trabalho personalizado na indústria da construção, a customização em massa surge como uma alternativa eficaz às estratégias atuais de pré-fabricação e padronização (Bianconi et al., 2019).

Esse artigo, então, apresenta testes de interoperabilidade a partir de um modelo de habitação de interesse social desenvolvido inteiramente no *Grasshopper*, para a transmissão de dados do par *Rhinoceros-Grasshopper* para os *software Archicad e Revit*, visando a comparação entre os principais *software* BIM do mercado.

O estudo analisa também o comportamento do modelo no *Archicad* e no *Revit* após a conversão, a partir de testes de modificações bidirecionais e atribuições de dados para a verificação de potencialidades e limitações no processo. Por fim, discute algumas estratégias e lógicas de modelagem algorítmica no *Grasshopper* para facilitar a comunicação com o BIM.

Esse trabalho objetiva, portanto, gerar um levantamento de estratégias atuais de conversão entre *software* de modelagem algorítmica e BIM, bem como definir algumas diretrizes de projeto a partir dos dois *software* escolhidos.

MATERIAIS E MÉTODOS

Este trabalho seguiu as seguintes etapas para a definição do método:

(a) Revisão bibliográfica sobre conceitos e temas como projeto paramétrico e algorítmico, BIM, interoperabilidade e customização em massa na Arquitetura.

(b) Levantamento de *software* e *plugins* para a transmissão de dados entre a plataforma de projeto algorítmico e uma plataforma BIM. Como opção de fluxo de trabalho para a transmissão de dados, para esta pesquisa adotou-se o modelo de importação através de *plugins*. A partir das análises realizadas, dois *plugins* foram eleitos para os testes de interoperabilidade, uma vez que se conectam aos dois *software* BIM mais utilizados atualmente: o *Archicad* e o *Revit*. Deste modo, a conversão foi totalmente executada no *Grasshopper* para o *Archicad* através do *plugin Archicad Live Connection* (ALC) e para o *Revit* através do *plugin Rhino Inside Revit* (RIR).

(c) Testes de interoperabilidade com o modelo de Habitação de Interesse Social (Gazel et al., 2018): Trata-se de um modelo algorítmico-paramétrico de unidades de habitação, com sistema estrutural em aço e peças padronizadas de mercado. Todas as entidades utilizadas na conversão como lajes, vigas, cobertura e paredes estão parametrizadas, o que facilita a conexão com famílias de atributos em ambiente BIM, bem como a transmissão de dados e atualização simultânea do modelo, em caso de modificações nos parâmetros.

Apesar de alguns *software* BIM possuírem nativamente possibilidades de modelagem algorítmico-paramétrica, como o caso do par *Autodesk Revit-Dynamo*, o ambiente *Rhinoceros-Grasshopper* foi adotado por ser um dos mais disseminados na atualidade.

Os *plugins* escolhidos (ALC e RIR) permitem que a geometria seja criada no *Grasshopper* de acordo com a linguagem, a descrição de componentes construtivos e configurações nativas do *Archicad* e do *Revit*, respectivamente. Estes foram os únicos *plugins* testados com conexão em tempo real com os *software* BIM, uma vez que operam bidirecionalmente, mantendo as propriedades intrínsecas do modelo *BIM*, sem a necessidade de parar a exploração formal, exportar a geometria e importar para o novo *software*. Segundo Guidoux Gonzaga et al. (2018) este processo representa uma continuidade no fluxo projetual dentro do ambiente digital, tornando-o mais eficiente em comparação aos demais *plugins* analisados, citados adiante.

Além disso, no caso de um projeto de customização em massa de habitação de interesse social, considera-se que a simultaneidade de visualização bidirecional permite aos usuários a possibilidade de modificar os parâmetros associados à geometria, bem como a aplicação de materiais, acabamentos, e famílias que reproduzem peças de mercado. A visualização simultânea do projeto, a documentação gerada (planilhas orçamentárias, quantitativos), dentre outras vantagens, permitem diretamente a aplicação de conceitos de customização em massa na Arquitetura, Engenharias e Construção.

FLUXOS DE TRABALHO

Visando adequar-se às tecnologias BIM, muitos escritórios de arquitetura têm implementado o uso do *Archicad*, do *Revit* ou do *Vectorworks*, sobretudo para documentação dos projetos. No entanto, para o desenvolvimento de geometrias complexas, destacam-se as ferramentas de modelagem algorítmica como *Grasshopper* (GH), Blender ou SideFX Houdini. Essas ferramentas de modelagem de fluxo de dados e procedimentos suportam a iteração de várias operações, possibilitando a criação rápida de geometrias complexas, enquanto os sistemas BIM são mais viáveis durante as fases posteriores, de desenvolvimento do projeto e documentação.

Kaushik (2017), argumenta que uma das principais barreiras à integração com o BIM desde o início do projeto é a divisão dos contratos em duas etapas sequenciais e não simultâneas: o projeto conceitual e o projeto detalhado. Após a concepção, normalmente outra equipe assume o controle (ou a mesma equipe), com o modelo, no entanto, reconstruído no *Revit* ou no *Archicad*. Nesse processo, muita inteligência paramétrica e associativa incorporada ao modelo conceitual é perdida ou desperdiçada para obedecer às limitações da ferramenta BIM.

Portanto, visando a integração em um mesmo projeto de sistemas algorítmicos e BIM, é importante considerar alguns fluxos de trabalho diferentes para a conversão de geometrias criadas no *Rhinoceros*-GH em um *software* BIM, como o *Archicad* ou o *Revit*. Cada tipo de importação possui suas vantagens e limitações:

Um dos fluxos de trabalho mais simples é a exportação de um arquivo .sat ou .dwg para o *Revit,* por meio de uma família de massa conceitual. No entanto, a geometria resultante dessa importação não pode ser editada, diferentemente dos elementos nativos do *software*.

Um segundo fluxo de trabalho é a importação por face. Nessa abordagem, é possível a criação de elementos nativos do *software* BIM, como o *Revit*, a partir de superfícies de elementos importados do *Rhinoceros*, utilizando-se os comandos "parede por face", "telhado por face", "pisos por face" e "sistema cortina por face". Esses elementos são hospedados na geometria *Rhino* importada e podem ser atualizados se a geometria base for alterada. No entanto, nem sempre o *software* BIM reconhece ou redefine o elemento *host,* sendo necessária a criação de novos elementos.

Um terceiro fluxo analisado, e com resultados mais promissores, são as conversões auxiliadas por *plugins*, melhor detalhadas a seguir.

CONVERSÃO ATRAVÉS DE *PLUGINS*

Os *plugins* de interoperabilidade para a conversão de dados de um *software* de modelagem algorítmica para o BIM propiciam melhores fluxos de trabalho, seja na criação de geometrias complexas, seja na iteração de várias operações para um controle mais preciso dos elementos BIM e suas propriedades. A figura a seguir ilustra os protocolos e conexões uni ou bidirecionais (representadas pelas setas) dos *plugins* testados na pesquisa, para a transmissão de dados do *software Rhinoceros* com as operações algorítmicas do *Grasshopper* aos *software* BIM *Revit* e *Archicad*.

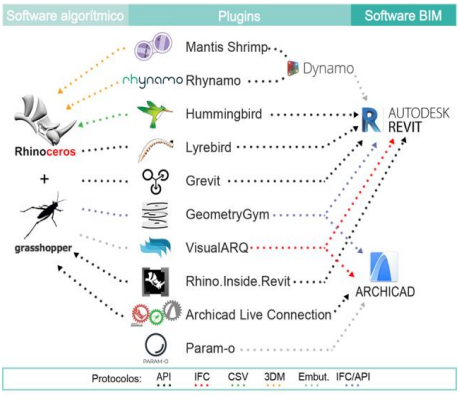

Figura 1 – Levantamento de fluxos de trabalho na transmissão de dados do *Rhinoceros* + *Grasshopper* ao *Revit* e ao *Archicad*, a partir dos diferentes *plugins* ativos testados, que serão detalhados a seguir. Fonte: Autores.

Janssen, Chen e Mohanty (2016) dividem esses fluxos de trabalho, diferenciando as suas relações de acoplamento em sistemas *Tightly Coupled* (fortemente acoplados) e *Loosely Coupled* (fracamente acoplados).

TIGHTLY COUPLED

Com a "abordagem fortemente acoplada", os sistemas são acoplados por meio da API (*Application Programming Interface*) fornecida pelo sistema BIM. Nesse caso, os sistemas baseados em grafos comunicam-se instanciando diretamente a geometria no modelo BIM cada vez que o modelo é executado (Janssen et al., 2016). Alguns exemplos são o *Autodesk Dynamo*, em conexão direta com o *Revit, que* utiliza sua API e o *Bentley GenerativeComponents*, que também utiliza a API, com conexão embutida com o *AecoSIM* e o *Marionette*, do *Vectorwork*. Eles são voltados principalmente para usuários que preferem permanecer no *software* BIM, com o qual estão familiarizados.

Recentemente, a *Graphisoft* incluiu o ***Param-o*** para a criação de objetos paramétricos no *Archicad* 24. Para a criação de objetos GDL, o *plugin* é intuitivo e eficiente, sobretudo quando já se tem noção da lógica projetual do *Grasshopper*. No entanto, para o desenvolvimento, como é o caso da pesquisa, de uma planta parametrizada, o *plugin* é limitado, uma vez que sua lógica é voltada à criação de objetos separadamente para a biblioteca e aplicação no *Archicad*, e não para criação de toda uma estrutura construtiva parametrizada de maneira integrada entre seus módulos, pilares, vigas, lajes e fechamentos.

Kaushik (2017) apresenta algumas limitações desses formatos e *plugins* embutidos nos *software* BIM: como os modelos BIM são, por sua própria natureza, grandes conjuntos de dados complexos, permitir que os usuários explorem parametricamente esses modelos pode reduzir drasticamente a latência e a robustez do sistema. Alterações paramétricas em modelos grandes podem tornar o *software* lento e muitas vezes resultar em erros inesperados, mesmo para usuários experientes. Além disso, os sistemas BIM já possuem interfaces de usuário muito complexas e a adição de recursos avançados de fluxo de dados e modelagem de procedimentos pode resultar em uma interface excessivamente complexa para ambos os casos de uso (Kaushik, 2017).

Nesse sentido, muitas ferramentas de interoperabilidade foram desenvolvidas para melhorar as transmissões de dados, visando a integração da geometria *Rhino-GH* com BIM. O GH possui *plugins* para conexão tanto ao *Archicad* (*Archicad Live Connection, Rhino GDL Converter*) como ao *Revit* (*Grevit, Hummingbird, Lyrebird, Rhino Inside Revit*). Existem também *plugins* que convertem geometrias *Rhino* para *Dynamo* (embutido no *Revit)*, como é o caso do *Rynamo*, e do GH para *Dynamo* através do *Mantis Shrimp*.

O ***Hummingbird***, por exemplo, exporta os componentes do GH em um arquivo de texto .csv que pode ser lido pelo *Revit*, para a exportação de suas geometrias. O arquivo de texto pode ser visualizado no *Hummingbird CSV-Viewer* (ou *Excel*) para estudo e edição de dados, caso necessário. Ao instalar esse *plugin*, seus componentes aparecem na guia "extra" do GH e na guia "*Add-ins*" do *Revit*, com o suplemento "*H Bird*". Esse *plugin* não visualiza a geometria no *Revit*, mas reconecta todos os dados necessários para, ao criar o arquivo .csv, reescrever o código, importando-o no *software*. Esse *plugin* possui conexão bidirecional com o *Revit*, é gratuito e de código aberto. No entanto, não possui conexão simultânea com o *Revit*.

O ***Lyrebird*** opera de modo semelhante ao Hummingbird, porém instanciando as geometrias através da API do Revit. Sua interface é muito simplificada, com apenas um

componente “LBOut” no GH, que possui *inputs* (dados de entrada) e *outputs* (dados de saída) com diversas funções. No *Revit*, há quatro comandos adicionados para gerenciar o *Lyrebird* e os elementos que ele cria. Esse *plugin* é gratuito e de código aberto. No entanto, não é bidirecional e não possui conexão simultânea. Além disso, não recebe atualizações desde 2014, o que o torna limitado em relação às correções de erros e às mudanças nos demais *plugins*.

O ***Grevit*** também converte as geometrias através da API do *Revit*, operando unidirecionalmente, de forma semelhante ao *Lyrebird*. No entanto, o *Grevit* possui parâmetros mais diversificados no GH, que podem ser definidos de acordo com as intenções projetuais, além de ser gratuito e com código aberto.

Por outro lado, a partir do crescente uso do ***Dynamo***, foram incorporados novos fluxos de trabalho que permitem aos usuários a troca de geometria e dados entre o *Rhino* e o *Revit* de modo mais eficiente do que a partir de *plugins* como *Grevit*, *Hummingbird* e *Lyrebird*, por exemplo. Com o ***Rhynamo***, os arquivos *Rhino* podem ser conectados às suas definições do *Dynamo* para desenvolver geometrias e controlar elementos do *Revit*. A ferramenta possui código aberto para leitura e gravação de arquivos *Rhino* *.3dm e permite que os designers criem fluxos de trabalho bidirecionais entre os *software*. Como principais característica positivas, o *plugin* é gratuito, de código aberto e capaz de uma rápida transposição de objetos para a plataforma do *Revit*. A complexidade de uso, além da necessidade de conhecimento das sintaxes do *Dynamo* e do GH, são os fatores mais limitantes.

O ***Mantis Shrimp*** (MS), à semelhança do *Rhynamo*, também possui código aberto e conecta o *Dynamo* (*Revit*) ao *GH* (*Rhino*). Ele permite a leitura direta de arquivos nativos do *Rhinoceros* (*.3dm), bem como a geometria de exportação do GH. No entanto, diferentemente dos demais *plugins* citados, que possuem uma interface visual, o MS é escrito em *Python* na forma de objetos de usuário (do GH para exportação) e em nós *Python* personalizados (do *Dynamo* para importação), o que exige maior conhecimento de programação.

A desvantagem de todas as soluções mencionadas é que elas só funcionam com ferramentas BIM específicas. A vantagem é a compatibilidade com o sistema BIM ao qual esses *plugins* se vinculam através de suas *APIs*, permitindo maior controle ao usuário. No entanto, o compartilhamento de modelos com outros profissionais será uma colaboração baseada em arquivo e não em nenhum padrão aberto.

LOOSELY COUPLED

Na “abordagem fracamente acoplada”, os sistemas são acoplados através da troca de modelos. O sistema baseado em grafos normalmente gera dados em um formato de arquivo padrão que pode ser importado diretamente no sistema BIM (Janssen et al., 2016).

Essa abordagem também possui *plugins* para o GH (*Geometry Gym* e *VisualARQ*) e para o *Rhinoceros* (*VisualARQ*). Todos utilizam o IFC como formato de troca, o que possibilita que os usuários criem seus próprios procedimentos personalizados de materialização.

O ***Geometry Gym*** é um *plugin* particularmente útil para estruturas 3D e análises estruturais. Ao instalá-lo, uma guia “GG” aparece no GH com diversos componentes do *plugin*. No *Revit*, é adicionada uma guia “*GeoGym*”. Assim como os demais *plugins*, em seu funcionamento baseado na lógica de formação de entidades no *Revit*, por exemplo, linhas conectadas corretamente aos *inputs* e *outputs* do *script* GG podem gerar colunas no *Revit*. Como limitações, tem-se o fato de ele não ser gratuito e não atuar simultaneamente com o *Revit*. Para atualizar qualquer informação ou geometria no arquivo no *Revit*, ele necessita ser importado novamente no formato IFC.

Em 2019, com o desenvolvimento do *plugin Rhino Inside Revit*, o *Geometry Gym* recebeu atualizações, com a inclusão de um novo componente “*ggRvt.BaketoRevit*”, que possibilita a conexão em tempo real com o *Revit*, utilizando sua API diretamente dentro do GH.

Por fim, o *plugin* ***VisualARQ*** possui ótimos recursos para documentação e também atua como um *add-on* do GH para programação visual e possibilidade de integração do BIM ao *Rhinoceros*. Alguns recursos, como as propriedades de materiais, por exemplo, são atribuídos somente no *Rhinoceros*, cujos arquivos são exportados em IFC. No geral, seguiu-se a mesma estrutura desenvolvida no *Geometry Gym* com algumas alterações de nomenclaturas e parâmetros. Recentemente, assim como o *Geometry Gym*, o *VisualARQ* também permite a complementação de recursos do *Rhino Inside Revit*, ou seja, a conectividade simultânea entre as ferramentas algorítmicas e BIM.

A vantagem de todos os *plugins* dessa abordagem é que eles funcionam independentemente do fluxo de trabalho, permitindo que os usuários vinculem ferramentas e sistemas para dar suporte a várias formas de colaboração e intercâmbio. No caso dos *plugins* que geram arquivos IFC padrão, os usuários têm a opção de se vincularem a qualquer aplicativo BIM que importe esse formato. No entanto, na prática, ainda existem muitos problemas com a implementação da IFC entre os principais s*oftware,* o que requer melhorias no padrão de implementações.

MODELO PARA TESTES DE INTEROPERABILIDADE

Com o intuito de comparar a exportação de um projeto gerado inteiramente no *Grasshopper* para o *Archicad* e para o *Revit*, foi utilizado um modelo de habitação de interesse social (Gazel et al., 2018). O modelo, denominado DOIS BITS, é parte de um trabalho sobre a reconstrução das habitações destruídas pelo desastre ambiental ocorrido no Brasil em Bento Rodrigues – MG, em novembro de 2015, após o rompimento de barragens de rejeitos de minérios na região, que deixou 226 famílias desabrigadas. f

Soma-se ao desastre ambiental, ainda, um cenário de déficit habitacional no país. O Brasil é alvo há décadas de diversas políticas públicas destinadas à produção habitacional (Cunha, 2016), mas ainda tem-se como resultado um déficit quantitativo e qualitativo de unidades.

> Em todo o país, novos bairros surgem em áreas distantes e sem urbanização, alinhando centenas de casas idênticas e minúsculas, ou enfileirando torres habitacionais com sofrível

> padrão construtivo, e grande impacto sobre o meio ambiente. Em face disto, a pergunta que nos vem naturalmente é: quais os resultados que essa produção provocará no cenário urbano brasileiro nos próximos anos? (FERREIRA, 2012, p.7)

Portanto, a inserção de tecnologias digitais nos processos de projeto possibilita a aplicação do conceito de customização em massa às novas habitações. Segundo Correia, Duarte e Leitão (2012), a personalização em massa permite modelos de alta qualidade a custos acessíveis, através de projeto e fabricação auxiliados por computador, o que reduz a repetição exaustiva dos processos. Ainda segundo os autores, essa abordagem supera um problema comum enfrentado pelos arquitetos ao lidar com grandes empreendimentos, e a dificuldade de projetar a variabilidade das unidades, com custo de construção sem os benefícios de economias de escala.

Além disso, a integração de processos de modelagem algorítmica e o BIM pode ser um caminho para ampliar a participação popular na tomada de decisões. Devido à facilidade de modificações nos *scripts*, é possível que os próprios moradores visualizem suas casas e definam suas prioridades habitacionais, organizadas sob a forma de parâmetros e regras aplicadas aos *scripts*.

Após o levantamento do perfil populacional e padrão construtivo das unidades unifamiliares do município de Bento Rodrigues, concebeu-se um projeto de habitação com sistema estrutural em aço, com o objetivo de facilitar o transporte e a rápida construção após o processo de fabricação. Replicado de acordo com regras predefinidas, os módulos dão origem à diversas unidades distintas, mas ao mesmo tempo compatíveis entre si - a unidade mínima resulta do agrupamento de 2 módulos, e a máxima de 5. Após estudos de organização espacial, adotou-se um módulo estrutural com dimensões de 4,80m x 6,00m (comprimento x largura). As superfícies de fechamento, por sua vez, são em sistema *steel frame* (placas OSB), cujas medidas mais comuns são 1,20m x 2,40m e 1,20m x 3,00m (base x altura).

Dentre as possibilidades do *script*, é possível modificar as dimensões de pilares e vigas, alturas dos pavimentos, medidas e quantidades de terças, inclinação, altura e posicionamento da cobertura, medidas dos módulos, suas quantidades e espaçamentos, disposição em formatos L, T, com segundo pavimento ou não, dentre outras possibilidades. Por outro lado, algumas restrições também foram impostas para seu correto funcionamento. As figuras 2 e 3 ilustram, respectivamente, os principais parâmetros modificáveis do *script* e algumas soluções formais encontradas a partir dele.

Quanto à construtibilidade do modelo, diversas possibilidades podem ser empregadas no *Archicad* ou no *Revit* após a exportação da unidade, no que se refere ao sistema estrutural (vigas e pilares), lajes, fechamentos externos e divisórias internas. A figura 4 ilustra o modelo testado de habitação com possibilidade mínima (2 módulos) após a renderização, já com os materiais aplicados.

A compartimentação interna é estabelecida pela posição em planta do núcleo rígido (banheiros, lavanderia e cozinha) e, portanto, o interior é concebido como um espaço flexível, que pode ser totalmente personalizável e adaptável ao longo dos anos.

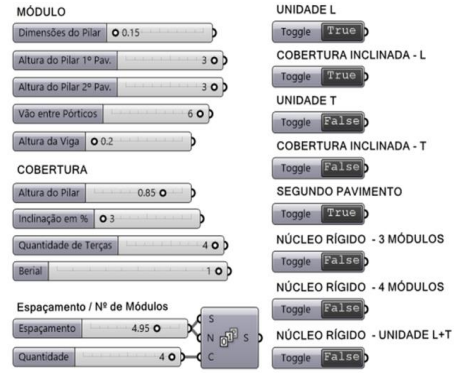

Figura 2 - Principais parâmetros modificáveis do script utilizado. Fonte: Autores.

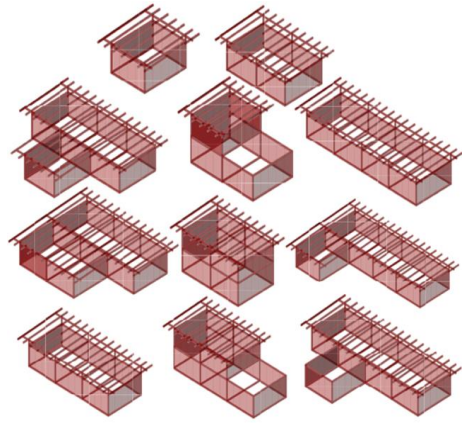
Figura 3 - Diferentes tipologias geradas a partir do script. Fonte: Autores.


Figura 4 - Modelo para testes renderizado, com aplicação de materiais e organização interna. Fonte: Gazel et al. (2017).

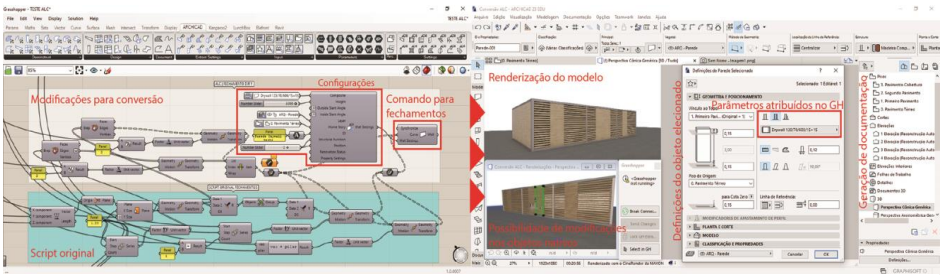

Figura 5 – Apresentação esquemática de parte da unidade modelada indicando o fluxo de trabalho entre GH e *Archicad* a partir do *plugin* ALC e a visualização do modelo com a aplicação de materiais para facilitar a compreensão dos usuários. Fonte: Autores.

PROCESSO DE EXPORTAÇÃO

Como visto, existem dezenas de *plugins* que se adequam em maior ou menor grau a diferentes necessidades projetuais. Após diversos testes, optou-se pela utilização de um *plugin* para importação no *Archicad* e outro para o *Revit*: *Archicad Live Connection* e *Rhino Inside Revit*, respectivamente. O principal diferencial de ambos é o fato de serem os únicos, dentre os testados, com conexão em tempo real com os *software* BIM.

A figura 5 ilustra o processo de conversão do modelo utilizado. Observa-se à esquerda a interface do GH com o *script* original, as modificações feitas para a conversão do modelo habitacional e algumas possibilidades de configuração dos fechamentos. À direita, tem-se a interface do *Archicad* com o modelo lido como objeto nativo BIM, possibilitando modificações em suas configurações no *Revit*, geração de documentação e geração do modelo renderizado.

Com o objetivo de exemplificar o processo de conversão do *software* de modelagem algorítmica para o BIM, será descrita resumidamente, a partir do *script* original, a lógica de formação das principais entidades utilizadas: pilares, vigas, fechamentos e lajes.

PILARES

Os pilares do *script* original foram feitos a partir da extrusão de um "*plane surface*" no eixo z, do piso ao segundo pavimento. Isso se repetiu movendo-se esse resultado a partir de vetores que posicionaram os pilares em suas posições corretas.

No ALC os pilares precisam ser estruturados a partir de dois pontos demarcando suas extremidades. Para tanto, a lógica foi simplesmente encontrar o centroide do plano inicial e de todas as movimentações de posicionamento dos demais pilares, bem como os pontos finais, ligando-os a uma pilha "*point*" para que sejam lidos pelo comando "*column*", do ALC. O componente "*column*" possui diversas possibilidades de configurações para os pilares, com relação a materialidade, dimensionamento, posicionamento, função estrutural, dentre outras propriedades.

No RIR os pilares são convertidos através do componente "*add column*", que recebe como *input* uma "*line*", que também pode ser obtida a partir de dois pontos. Portanto, assim como no ALC, a lógica aplicada foi obter o centroide do "plane surface" no piso e", movê-lo à altura do pilar no primeiro pavimento, utilizando os dois pontos para gerar uma "*line*" e ligá-la ao "*add column*". A partir da geração do primeiro pilar, pode-se replicá-lo para as demais posições. O componente "*add column*" do RIR possui *inputs* que possibilitam selecionar o tipo de elemento (tipo de pilar usado, escolhido no Revit, e posteriormente ligado a uma categoria de modelo) e um seletor de níveis para os documentos.

VIGAS

No script original, as vigas foram geradas de forma análoga aos pilares: através de extrusões de planos, porém nos eixos x e y. Essas extrusões foram reproduzidas com comandos "*move*" e "*series*" para obtenção de todos os trechos de vigas.

Para serem passadas ao *Archicad*, o comando "*beam*" do ALC recebe *inputs* através de curvas (ou linhas), e não através dos volumes extrudados. Novamente, as linhas foram criadas a partir dos centroides inicial e final. A mesma lógica foi aplicada às demais vigas, modeladas a partir de um comando "*line*" no componente "*beam*". Esse componente possibilita configurações de materias para as vigas, dimensões, camadas (ou vegetais, como são chamadas no *Archicad*), ID, função estrutural, posicionamento, dentre outras propriedades.

O RIR teve a mesma lógica para a conversão dos pilares, através do componente "*add beam*". Assim como para os pilares, o comando de vigas possui diversas possibilidades de configurações, e possibilita selecionar o tipo de elemento (tipo de viga usada, escolhida no *Revit*, e posteriormente ligada a uma categoria de modelo) e um seletor de nível de documentos.

FECHAMENTOS

O *script* original gerou painéis a partir de "*plane surface*", unidos por comandos "*merge*" e replicados através de "*move*", ligados a expressões simples de soma de vãos com espessuras dos pilares e "*series*".

No ALC, diferentemente da lógica utilizada no *script* original do GH, o comando "*wall*" para criação de paredes recebe como *input* as 4 arestas em vez do plano, através de "*curve*" ou "*line*". Para essa conversão, os planos foram descontruídos em arestas para a conexão ao comando "*wall*" como curvas. Esse comando permite modificar configurações de materialidade, dimensionamento, posicionamento, ID, função estrutural, dentre outras propriedades.

Já no RIR, o comando "*add wall*" para criação de paredes recebe como *input* apenas uma aresta da parede, através novamente de uma "*curve*" ou "*line*". Assim como no ACC, os planos foram decompostos em arestas para a conexão ao comando "*wall*" como "curve". No entanto, para simplificar o *processo*, em vez de gerar no GH vários painéis de 1,20m x 3 m cada, representando as dimensões padrões de mercado para placas OSB para estrutura em *steel frame*, optou-se por criar a linha inferior na dimensão dos módulos (de 6 m), que posteriormente poderão ser divididos em painéis com as dimensões desejadas no próprio Revit, ao especificar o material utilizado.

O componente "*add wall*" do RIR possibilita configurações do tipo de fechamento usado, escolhido no *Revit*, do nível, da altura da parede e do posicionamento da linha de eixo. Permite também a inversão de faces interna e externa, a união das extremidades e a seleção entre parede estrutural ou não de acordo com os esforços atribuídos.

PISOS E LAJES

Como o *script* original não possuía lajes, optou-se por criá-las para complementar o projeto, utilizando-se a mesma lógica dos fechamentos, a partir de planos, séries e movimentações para posicionamento correto.

No ALC, o comando para lajes do ACC "*slab*" recebe como *input* polígonos, criados pelo comando "*geometry*". Para transformar as lajes nessas geometrias, foi utilizada a mesma lógica dos fechamentos, descontruindo-se os planos em arestas para a conexão. O componente de lajes pode ser ligado ao "*slab settings*", permitindo modificações de materias, dimensões, ângulo de referência, camadas (vegetais), ID, função estrutural, posicionamento, dentre outras propriedades.

Já o componente para lajes do RIR "*add floor*" recebe como *input* um "*boundary*", através de uma "*curve*" ou "*surface*" por exemplo. Isso possibilitou ligar diretamente os planos da laje, replicando-os e movendo-os para o correto posicionamento em ambos pavimentos. Esse componente de paredes do RIR pode ser configurado com relação ao tipo de fechamento usado, ao nível, e à função, sendo estrutural ou não.

RESULTADOS PRELIMINARES

Após as conversões realizada em ambos *plugins*, foram analisadas suas potencialidades e limitações. Convém destacar que algumas análises são subjetivas e dependem da familiaridade com cada *software* BIM e sua interface.

Em ambos os casos, todas as entidades geométricas foram criadas a partir de pontos, curvas ou superfícies, o que está diretamente relacionado à lógica de desenho em cada *software* BIM. No *Revit*, por exemplo, as paredes são desenhadas a partir de linhas, tornando-se também a geometria utilizada no GH para leitura do *plugin* de conversão do RIR.

Com relação às interfaces, considerou-se a interface do RIR mais completa, uma vez que apresenta maiores informações a serem atribuídas a cada elemento, além da organização em listas para escolha de dados, semelhante a uma extensão do *Revit* dentro do GH. A interface do ALC no GH tampouco é complexa, mas utiliza pouco as listas, o que pode torná-la menos intuitiva. No entanto, o ALC consome menor tempo de processamento, tornando a conexão instantânea entre *software*. O RIR, por sua vez, demanda um dispêndio computacional maior, o que exige uma máquina com capacidade de processamento mais robusta.

Para o RIR, as medidas das seções transversais do *script* original dos pilares, vigas, fechamentos e lajes são desnecessárias no modelo final exportado, uma vez que o *plugin* utiliza os atributos e métricas relacionados às famílias de entidades disponíveis no *Revit*. Todas as famílias de objetos exibidos no modelo no *Revit*, tornam-se opções listadas também no GH.

O *Archicad*, por sua vez, permite a utilização das dimensões do *script* original ou a escolha de qualquer entidade (laje, viga, pilar, etc) existente como famílias no *Archicad*.

No *Archicad* também é possível escolher se a entidade será gerada no eixo central, topo ou base da seção transversal, o que facilita a junção entre pilares e vigas, por exemplo. No *Revit*, no entanto, as entidades são sempre geradas a partir do eixo central da geometria, exceto os planos de paredes.

Caso as entidades sejam desbloqueadas no *Revit*, as alterações posteriores no *script* do GH não são computadas neste *software* BIM. No entanto, observou-se um maior controle dessas alterações no *Archicad* após o desbloqueio.

O modelo construtivo resultante no *Archicad* e no *Revit* pode ser aprimorado em ambas interfaces após suas conversões. Para facilitar a compreensão por parte dos usuários, é possível produzir imagens tridimensionais que indiquem acabamentos e detalhes construtivos. Com a documentação gerada, também pode-se obter uma estimativa da quantidade e dos custos dos materiais de construção.

A possibilidade de usar componentes diferentes depende da biblioteca de objetos que cada usuário possui, tanto para o *Archicad* quanto para o *Revit*. No *Revit*, para o carregamento de novos elementos, deve-se "recomputar", para que eles apareçam nas listas do GH. Porém, cada vez que se recomputam os dados, o RIR cria elementos de revisão duplicados no *Revit*. A mesma limitação de sobreposição ocorre ao abrir um arquivo salvo no GH e outro no *Revit*. Isso inviabiliza processos primordiais, como por exemplo a utilização de novas famílias adicionadas ao *Revit*. Ao abrir somente o arquivo salvo do GH, essas famílias do *Revit* não são carregadas na lista do GH.

Portanto, essa limitação inviabiliza o uso eficiente do *plugin* RIR. Ao contestar os desenvolvedores sobre esse problema em um fórum de uma plataforma colaborativa, eles garantiram que estão tentando corrigir desde fevereiro de 2020, e espera-se que, em breve, essa falha seja solucionada.

A partir dos resultados expostos, embora o RIR tenha um enorme potencial, considerou-se o ALC mais eficiente até o momento para as transferências de dados entre o GH e o *Archicad,* como um *software* BIM.

DISCUSSÕES E TRABALHOS FUTUROS

A decisão sobre a transição entre plataformas geralmente resume-se à necessidade de transferência de dados. No entanto, de acordo com o objetivo do projeto, em determinados momentos é válido importar geometrias sem o consumo de tempo para traduzir os dados atribuídos a estas. Em outros, é importante criar geometria e dados nativos para utilizar as ferramentas de documentação de *software* BIM ou permitir a edição do modelo após sua inserção. Para projetos que exigem geometria nativa, é importante adotar o método mais eficiente para percorrer as plataformas de *software,* sem o retrabalho de modelagem da geometria e inserção de dados.

Nesse cenário, os campos de projeto e construção tem se dedicado a alternativas para os problemas de interoperabilidade e compatibilidade de *software*. Ao longo da pesquisa, observou-se que todas as soluções discutidas estão continuamente em modificação para atualização e correção de erros em *plugins*, seja a partir de discussões em plataformas colaborativas, em iniciativas *open source* ou de *software* proprietários.

Por sua vez, o recurso de portabilidade dos *plugins* do *Grasshopper* possui algumas limitações. Em diversos casos, partes significativas do *script* não são portáteis e várias operações são refeitas para a geração da geometria em diferentes *software*. Neste sentido, apesar dos diversos *plugins* analisados possuírem diferentes interfaces e protocolos para transferência dos dados, nenhum é perfeitamente eficiente em todos os aspectos. As barreiras de interoperabilidade devem-se em parte às limitações da API do *software* BIM e também à falta de desenvolvimento do formato de arquivo IFC, que busca interoperabilidade universal no setor de AEC.

Adicionalmente, há barreiras culturais para a utilização integrada de diversos *software* em um mesmo modelo de projeto. Como consequência, há reconstrução das geometrias em *software* BIM, que além do retrabalho, resultam na perda de grande parte da inteligência incorporada ao *design* original (Kaushik, 2017). Neste caso, o resultado final pode retroceder às limitações de modelagem do *software* BIM*,* em um processo que contradiz os princípios conceituais da modelagem da informação da construção.

Nesse contexto, como trabalhos futuros, objetiva-se aprofundar os estudos sobre a interoperabilidade do projeto algorítmico ao BIM e o seu potencial de aplicação em projetos de customização em massa na arquitetura. Pretende-se analisar também a utilização de nuvens no processo projetual, possibilitando o trabalho compartilhado e integrado através de computadores em diferentes localidades.

Por fim, tendo em vista o projeto de unidades de habitação de interesse social, vislumbra-se que, com o domínio e integração das plataformas, os arquitetos possam criar interfaces personalizadas e intuitivas de *design* arquitetônico interativo. Essas ferramentas permitirão aos usuários aplicar regras para personalizar seus interesses projetuais, dentro de possibilidades codificadas por um modelo predefinido, bem como especificar dimensões e acabamentos que possam atender às suas restrições econômicas.

AGRADECIMENTOS

Os pesquisadores agradecem a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro à pesquisa, ao Laboratório de Modelagem Digital Nó.Lab e ao Programa de Pós-Graduação em Arquitetura e Urbanismo da UFV.

REFERÊNCIAS

Bianconi, F., Filippucci, M., & Buffi, A. (2019). Automated design and modeling for mass-customized housing. A web-based design space catalog for timber structures. *Automation in Construction*, *103*(March), 13–25. https://doi.org/10.1016/j.autcon.2019.03.002

Castelo Branco, R., & Leitão, A. (2017). Integrated algorithmic design: A single-script approach for multiple design tasks. *ECAADe 35 - Design Tools - Theory*, *1*, 729–738. https://doi.org/10.1063/1.1649724

Correia, R., Duarte, J., & Leitão, A. (2012). GRAMATICA: A general 3D shape grammar interpreter targeting the mass customization of housing. *Digital Physicality - Proceedings of the 30th ECAADe Conference - Volume 1 / ISBN 978-9-4912070-2-0, Czech Technical University in Prague, Faculty of Architecture (Czech Republic) 12-14 September 2012*, *1*(Knight), 489–496.

Cunha, T. F. (2016). *Editais , contratos e medições do Programa Minha Casa Minha Vida (PMCMV) com base nos critérios de desempenho da norma NBR 15 . 575 / 2013 da Associação Brasileira de Normas Técnicas (ABNT)*.

Feist, S. T. de V. (2016). *A-BIM : Algorithmic-based Building Information Modelling*. *May*.

Ferreira, J. S. W. (Coord. . (2012). Produzir casas ou construir cidades? Desafios para um novo Brasil urbano. Parâmetros de qualidade para a implementação de projetos habitacionais e urbanos. In *Surveillance and Society* (Vol. 2, Issues 2–3).

Gazel, J. L., Martinez, A. C. P., dos Santos, D. M., & Lopes de Souza, D. (2018). *2 BITS: A case of mass customization for social housing*. 353–358. https://doi.org/10.5151/sigradi2018-1744

Guidoux Gonzaga, M., Prazeres Veloso de Souza, L., Paiva Ponzio, A., & Miotto Bruscato, U. (2018). *Cutting the Path: Encouraging Formal Exploration Through Integration Between Algorithmic and BIM Environments*. 11–16. https://doi.org/10.5151/sigradi2018-1275

Janssen, P., Chen, K. W., & Mohanty, A. (2016). Automated Generation of BIM Models. *ECAADe*, *2*, 583–590. http://papers.cumincad.org/cgi-bin/works/Show?ecaade2016_239

Kaushik, V. (2017). *Why do Architects need Computational BIM Workflows?* https://wowad.in/why-do-we-need-computational-bim-workflows/?utm_campaign=Thank God It%27s Computational&utm_medium=email&utm_source=Revue newsletter

Larrondo, A. (2017). Generación y control de formas libres en entornos BIM : modelado paramétrico, modelado algorítmico. *TDX (Tesis Doctorals En Xarxa)*. http://www.tdx.cat/handle/10803/457875

Schumacher, P. (2016). *Design Parameters to Parametric Design*. http://www.patrikschumacher.com/Texts/Design Parameters to Parametric Design.html