AS226214

Custom Computational Workflows for BIM Design Implementation

Eckart Schwerdtfeger
BIM Associate | Zaha Hadid Architects

Learning Objectives

- Understand how ZHA uses innovative custom workflows to implement BIM models for their signature fluid and parametric architecture
- Effectively automate cross-application workflows using custom tools, visual scripting and the cloud
- Create parametric freeform elements inside Revit + Dynamo using different methods like forms, adaptive components and T-Splines
- Develop efficient facade workflows utilising computational design, intelligent adaptive BIM elements and meta data analysis

Description

Zaha Hadid Architects are known for their signature fluid architecture and computational design. The transformation of these projects into Revit BIM models represents a special challenge and requires custom workflows.

During the last years, we have created BIM models for a number of iconic global projects and have used a variety of efficient and innovative methods that could enhance and inspire your work as well.

This presentation will explore practical examples that deal with the efficient cross-application transfer of geometry and metadata using custom software, visual scripting and the cloud. We will show how to utilise the different types of Revit families to generate parametric, rule-based geometry and freeform elements.

How can visual scripting in Dynamo enhance our computational design and how could an efficient façade workflow utilising computational design, intelligent BIM elements, and metadata analysis look like?

What challenges did we face and how did our methods evolve?

Speaker

Eckart Schwerdtfeger is an architect with 12 years of experience in all project phases and in implementing large-scale, high-profile international projects in Europe, Russia, the Middle East and China.

During his time at Behnisch Architects, Coop Himmelb(l)au and LAVA he parametrically designed, optimised and implemented several remarkable facades. His main fields of work were computational design, BIM management and programming.

Currently, Eckart is BIM Associate at Zaha Hadid Architects and responsible for managing the office's BIM team. Besides leading and implementing several large-scale BIM projects, he focuses on developing the ZHA global BIM workflows.

He develops and writes proprietary software, plugins and scripts to effectively facilitate, optimise and automate BIM and cross-application tasks and to enable the design team to contribute to the BIM models more directly and fluently.

Index

Introduction Aims

During the course of an architectural project, from concept design to documentation and implementation, a lot of different software is used. This e.g. depends on the type of project, the architect's scope, requirements and deliverables, even the skills and preferences of the individual team members.

Additionally, the design will change regularly, either due to testing of various options, to optimise and value engineer the initial design or to implement coordination changes and details in later phases.

To translate a design model into a proper BIM model and to coordinate both is a particular challenge. The design team should be enabled to contribute to the BIM model in a very direct and fluent way to save time and prevent issues.

Our main aims are:

Effective cross-application workflows from design to BIM

- Be able to use content created in 3rd party software
- Prevent duplicated work

Parametrise and automate as much as possible

- Enable + accelerate creation of design iterations and implementation of changes
- Save time spent on repetitive tasks

To facilitate this we use

Custom...

- BIM Elements
- Visual Scripts / Scripts / Macros
- Software / Plugins

Introduction Customisation Options

Every software provides a set of general tools that can be used to e.g. create a BIM model. However, this might not be enough for every office, certain projects, workflows or tasks that require special or custom solutions.

Fortunately, most software provides various possibilities to customise and extend it's base functionality, e.g. a visual scripting environment, macros or application programming interface [API].

Introduction Customisation Options

Customisation options for Revit + Dynamo:
[sorted by difficulty]

Revit

Family	Modelling Parametric geometry + conditions + formulas Revit family editor
Macro	Scripting [e.g. C#, VB.Net, Python, Ruby] Access to Revit programming interface Integrated editor, SharpDevelop
Plugin	Programming [e.g. C#, VB.Net] Standalone library with advanced capabilities External editor, e.g. Visual Studio

Dynamo

Visual Script Custom Node	Visual Scripting Graphical access to the programming interface Dynamo
Script	Scripting [e.g. DesignScript, Python] Access to Dynamo + Revit programming interface Integrated script nodes
	Programming [e.g. C#, VB.Net] Standalone library with advanced capabilities External editor, e.g. Visual Studio
Zero Touch Node Custom Node	Simple user-defined functions Custom user interface possible
Extension	Global access and extension of Dynamo possible

Forge

The Autodesk Forge cloud platform can be used for design automation as well. It currently operates on AutoCAD / DWG files but in 2019 Revit BIM models will be supported and can be automated, too.

- Master plan with extent of about 800 x 300 m
- Around 40 buildings with varying size and typology

Aims

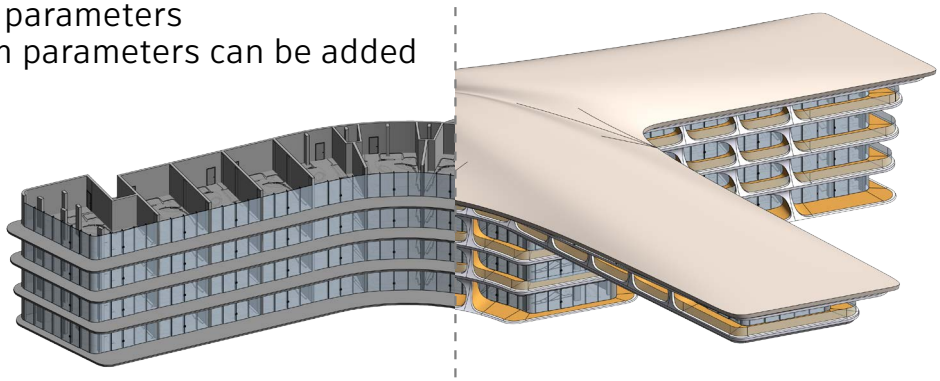
- Parameterisation and automation of the design process
- Effective use of freeform geometry in BIM

Native Elements

- E.g. floors, walls, columns, beams, curtain walls, doors, furniture, etc.
- Preset constraints and restrictions for every category
- Preset parameters
- Custom parameters can be added

Creation

- Manual drawing
- Manual conversion of pre-existing CAD lines
- Automated custom workflow



Native Elements

For some buildings the manual drawing or conversion process might not be efficient enough to create a BIM model, e.g. due to scale of building, number of elements or design in general.

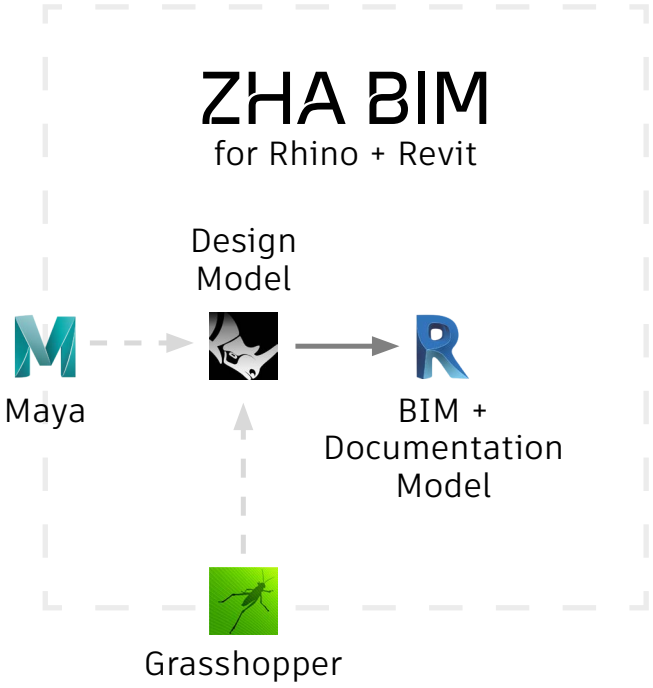
An automated process could be a much better or only solution, especially when the building elements are already available in some format or software and the design changes regularly and in quick succession.

Example shown in Talk + added afterwards

ZHA BIM

Various applications [e.g. Maya, Grasshopper, Rhino] are used for designing. Those different models are all collated in a Rhino Design Model which also acts as base for the Revit BIM + Documentation Model.

ZHA BIM was developed to efficiently translate geometry and meta data between the Design and BIM model and contains plugins for Rhino + Revit.



To Transfer an element cross-application we need

Geometry
of element or it's underlying geometry

Instructions
for transfer process and targeted software

Building element information
Parameters, meta data [optional]

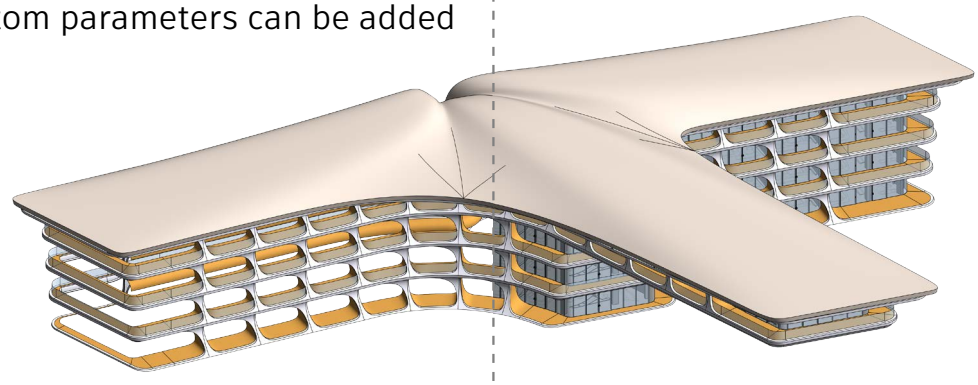
Example shown in Talk + added afterwards

Freeform Elements

- E.g. sculpted roof, facade and balcony cladding, balustrades, etc.
- Freeform geometry not supported by preset building elements
- Custom parameters can be added

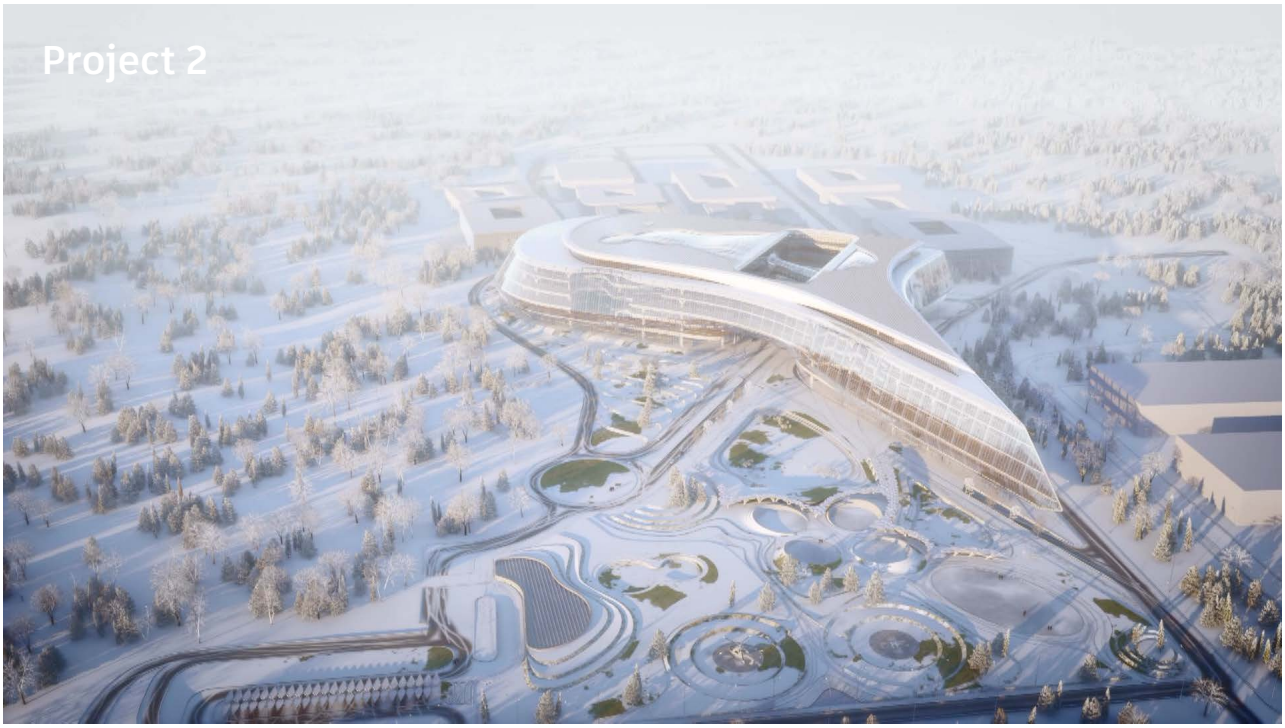
Creation

- Manual import [tedious process, slow, limitations]
- Automated custom workflow



Standard Revit building elements do not support freeform geometry. This elements need to be loaded into generic model families and processed afterwards to enable most of the Revit features. This is a very manual and tedious process, hence an automated workflow was required.

Example shown in Talk + added afterwards



- Building complex of about 262.000 m^2
- Office building for up to 17.000 employees working in innovation + development

Aims

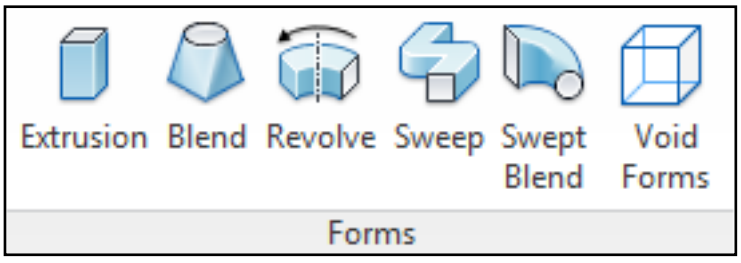
- Parametric Revit elements instead of importing “dead” geometry
- Intensified use of visual scripting / Dynamo

Freeform Elements In Revit

Importing freeform geometry with the ZHA BIM plugin is very fast and simple. However, the resulting elements are “dead” and cannot be edited inside the BIM model.

Revit is able to create certain rule-based geometry natively and parametrically. It takes a bit longer to set this up but in the course of the project changes can be implemented directly and very efficiently.

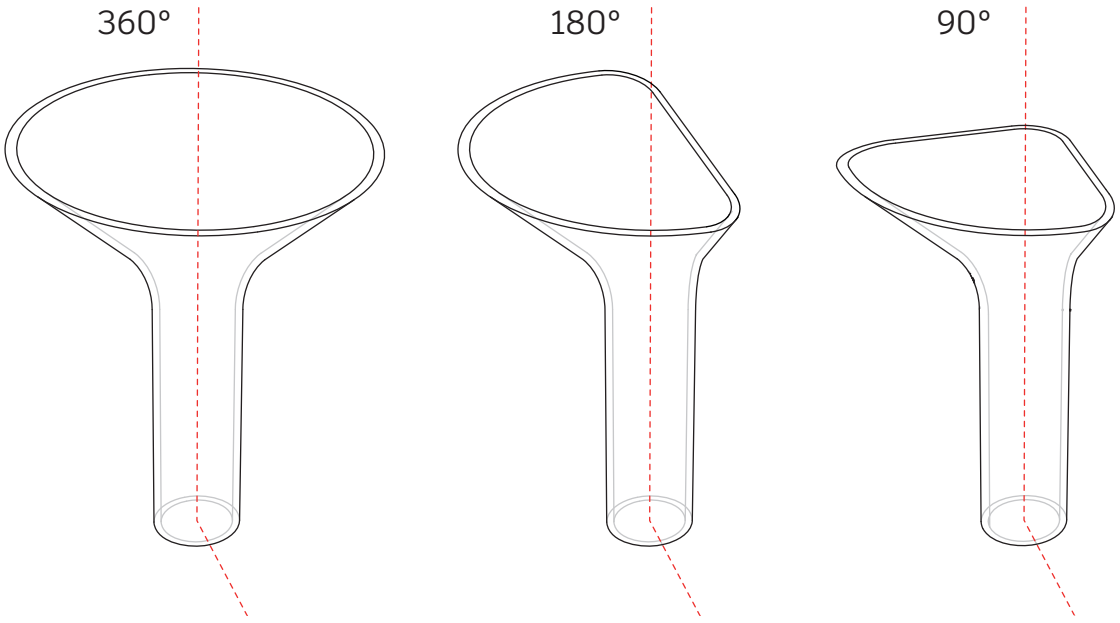
Revit provides the following methods for creating forms:



Freeform Elements In Revit Forms

Rather complex forms can be created inside Revit parametrically in case they can be broken down into individual pieces that follow rules that Revit understands.

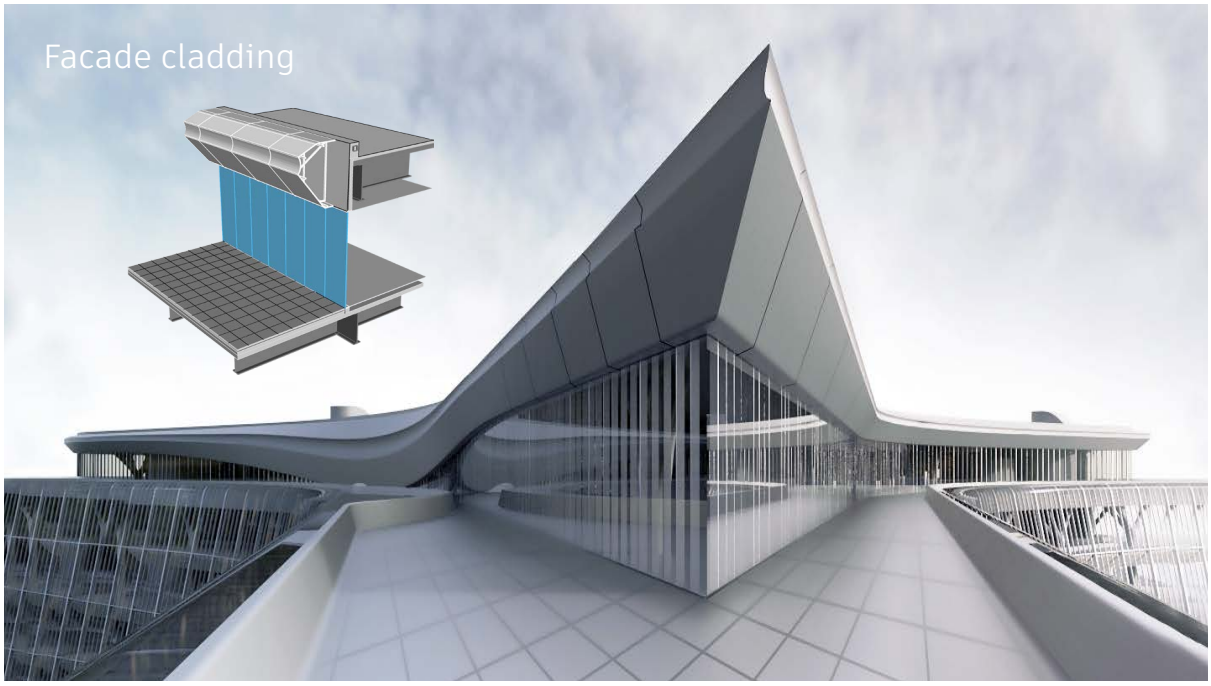
Mushroom column cladding



Example shown in Talk + added afterwards

Freeform Elements In Revit Adaptive

Adaptive families prove to be very useful for generating rule-based freeform elements as well. This facade cladding was originally modelled in Rhino and then translated into a proper and fully parametric Revit element for added efficiency and simple changes.



Example shown in Talk + added afterwards

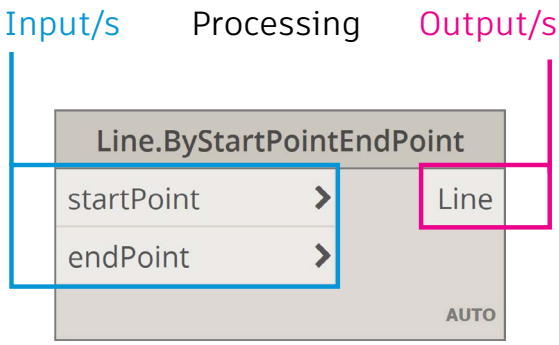
Dynamo Visual Scripting

Dynamo provides a graphical user interface for controlling the Dynamo + Revit application programming interface [API].

With visual scripting custom workflows and processes can be created rather easily and without having to learn a programming language first.

Each script represents a linear sequence of nodes. A node can receive geometry or data via one or multiple input/s, processes them in a predefined way and generates one or multiple output/s.

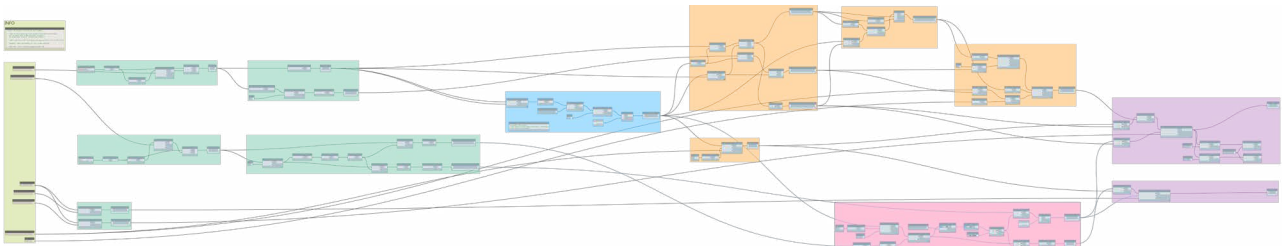
Inputs and outputs are connected using wires and this defines the sequence of commands that is run.



Tree Column Structure

Dynamo is a Revit plugin, hence the open building model is fully accessible, can be analysed and processed in a Dynamo script.

E.g. the structure for the tree columns can be generated based on the live building model and is automatically correct. This is difficult to achieve if multiple 3D models exist and have to be manually coordinated regularly.



Example shown in Talk + added afterwards

Dynamo Visual Scripting T-Splines

Dynamo has the ability to translate low poly meshes into smooth freeform geometry using T-Splines. The resulting Revit family is not editable directly but the tedious process of importing 3rd party geometry is no longer required and the result is usually much cleaner.

This process was used for the 400 custom tree columns in the building

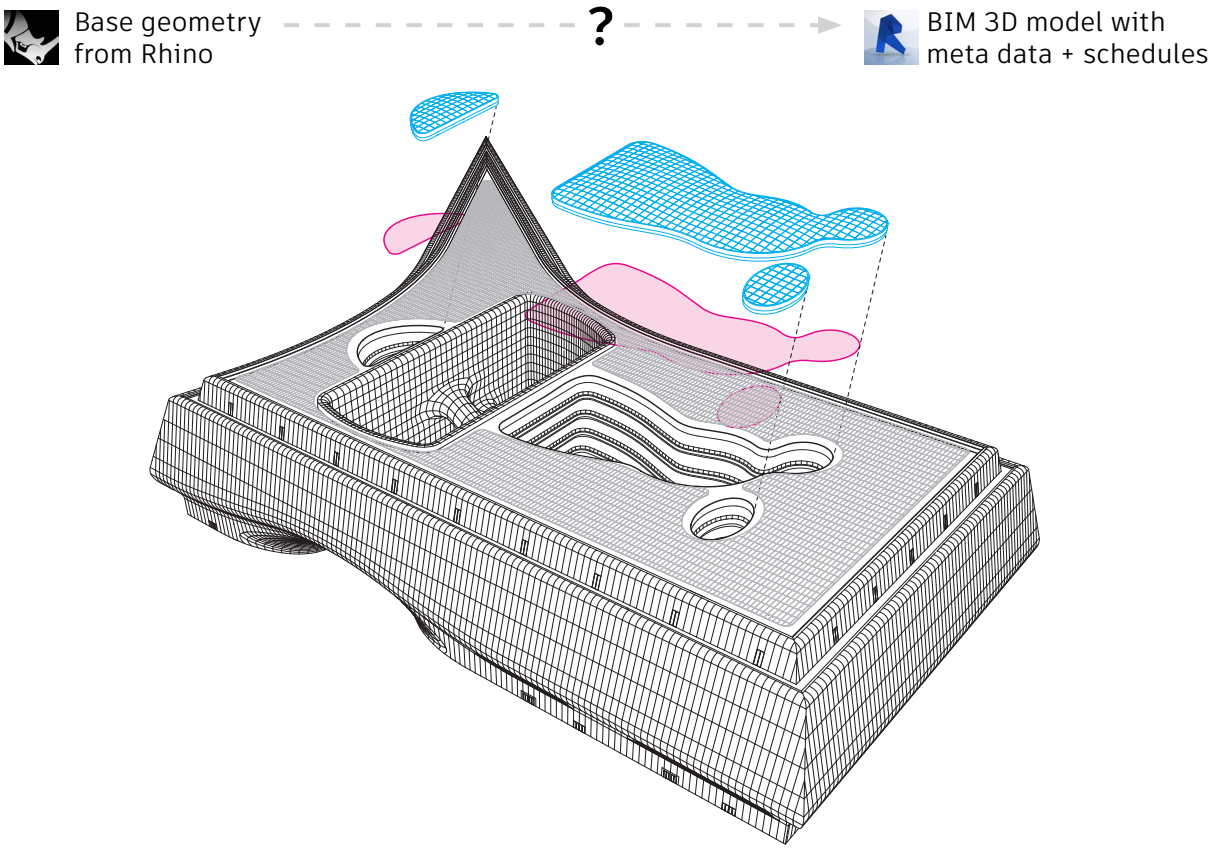
Example shown in Talk + added afterwards

Dynamo Visual Scripting DynaShape

DynaShape is a Dynamo package for constraint-based form finding, optimisation and physics simulation by Long Nguyen.

It allows a user to model complex curved forms that are based on certain physical or structural types (e.g. catenary shells, tensile structures) or that meet certain geometric properties (e.g. planar quad panels, unrollable surface stripes, as-uniform-as-possible edge length, etc.).

DynaShape was used for the atrium skylights to generate a structurally sound design in a process reminiscent of Gaudi's hanging chain models.



Example shown in Talk + added afterwards

Dynamo Visual Scripting Custom Nodes

Dynamo can be extended via custom nodes and many different methods are provided to facilitate this.

Visual scripts can be packaged into custom nodes right inside Dynamo. If advanced capabilities are required a standalone editor like Visual Studio and a programming language like C# are to be used.

Zero Touch Nodes are the easiest to program and implement, while custom nodes can contain an individual user interface as well. Additional knowledge of the Windows Presentation Foundation [WPF] graphical subsystem is required to use this properly.

A number of custom user interface nodes was developed with the aim to research, develop, innovate and to create efficient, intelligent and adaptable nodes.

Example shown in Talk + added afterwards

Dynamo Visual Scripting Extensions

Custom nodes can integrate new functionalities and user interfaces within Dynamo. Since they are part of a script and run in a self-contained way, their access is limited to their own context and their capabilities are limited.

Extensions are plugins that gain access to the entire Dynamo environment, e.g. the Dynamo user interface, display and open scripts. They can be used to add to or control Dynamo on a global level.

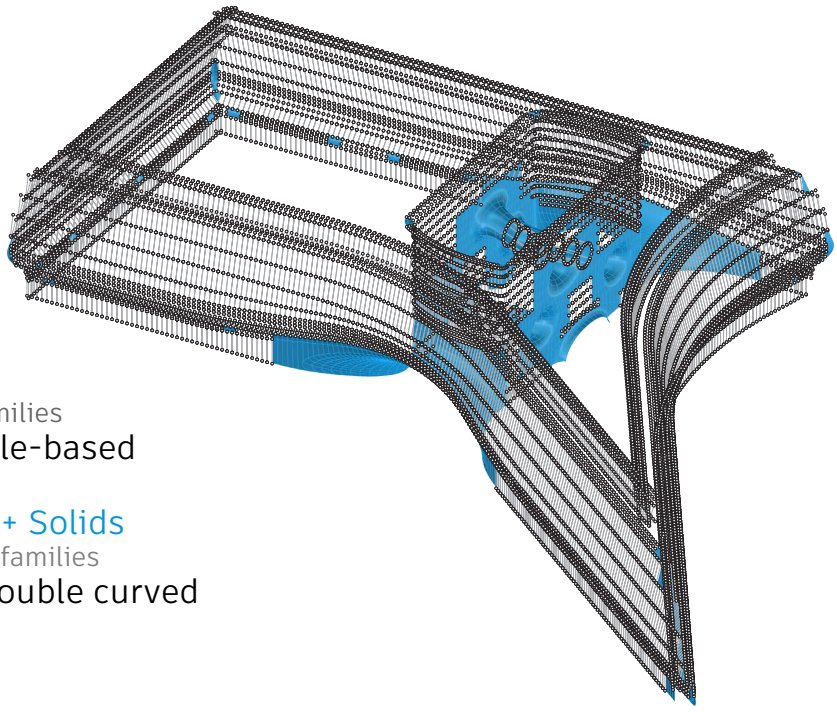
Using a ViewExtension plugin the previous research and developments can now be applied to the out of the box Dynamo nodes as well.

Example shown in Talk + added afterwards

Adaptive Elements

Adaptive elements are parametric families that can be placed and controlled just by points and meta data [very lightweight and easy to transfer]. This makes them very efficient for implementing flat or single curved facade or roof panels within a three-dimensional space.

A pre-existing Grasshopper script was able to provide the points and meta data required for instantiating the rationally-shaped panels. Double curved panels had to be transferred as geometry imports.



Point cloud
for adaptive families
Rational / rule-based

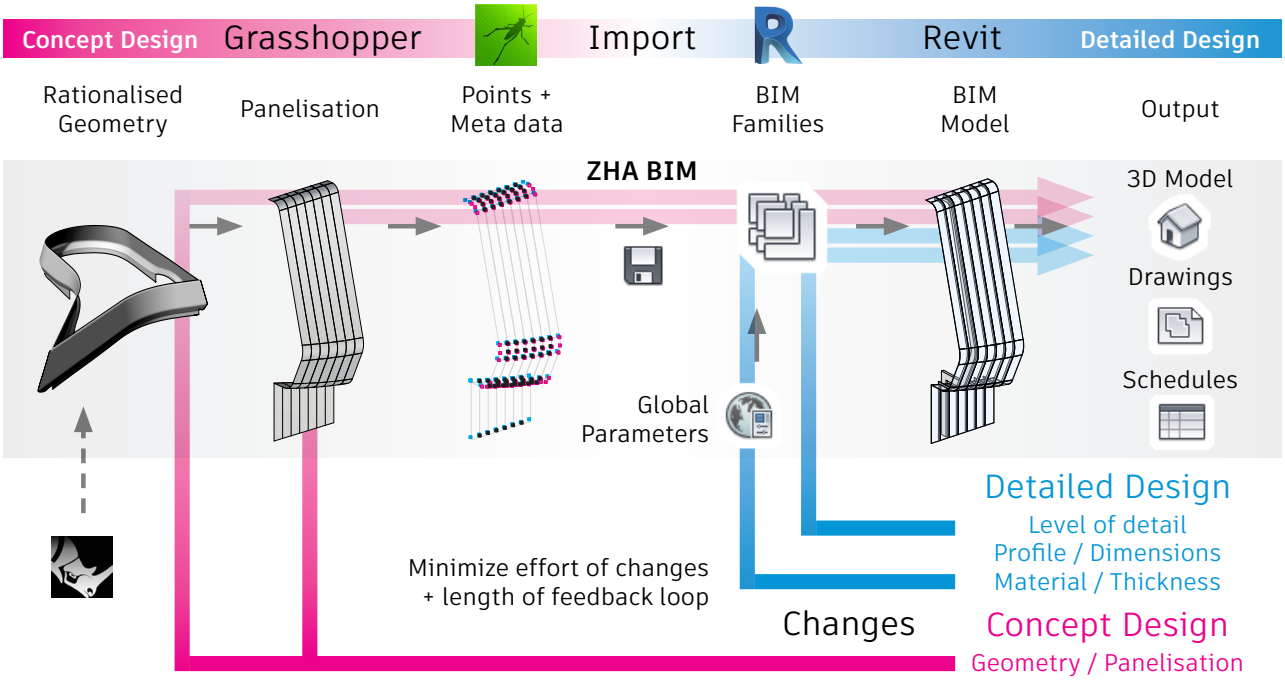
3D Surfaces + Solids
for form-based families
Freeform / double curved

Adaptive Elements

The aim was an efficient workflow from Grasshopper to Revit that enables regular design changes and fast adaptations of the BIM model. ZHA BIM was used to transfer the required coordinates and meta data from Rhino to Revit and to instantiate the panels.

During concept design phase the base geometry and panelisation changed regularly, hence the automated translation process needed to be run completely.

In detailed design phase changes were limited to level of detail, profiles, dimensions, etc. and enabled via parametric families and global parameters. This minimises the effort of changes and feedback loop length. Rhino and Grasshoper are taken out of the process and the focus fully shifts to the parametric BIM model.



Example shown in Talk + added afterwards

ZAHA HADID ARCHITECTS

Studio London
10 Bowling Green Lane
London EC1R 0BQ
T +44 20 7253 5147
F +44 20 7251 8322

mail@zaha-hadid.com

www.zaha-hadid.com
www.twitter.com/ZHA_News
www.linkedin.com/company/zaha-hadid-architects/

Eckart Schwerdtfeger
BIM Associate

Eckart.Schwerdtfeger@zaha-hadid.com
Eckart.Schwerdtfeger@gmx.net

www.parametrised.net
www.twitter.com/Eckart_S
www.linkedin.com/in/eckart-schwerdtfeger/